The Report Committee for Suriya Gunasekar Certifies
that this is the approved version of the following report:

A Survey on using Side Information in Recommendation Systems

APPROVED BY

SUPERVISING COMMITTEE:

Joydeep Ghosh, Supervisor

Sujay Sanghavi

A Survey on using Side Information in Recommendation Systems

by

Suriya Gunasekar, B. Tech

REPORT

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

MASTER OF SCIENCE IN ENGINEERING

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2012

Dedicated to my parents Mrs. Shantha, P. and Mr. Gunasekar, S.

Acknowledgments

I take pleasure in thanking everyone who have been instrumental in the successful completion of this report. I sincerely thank my supervisor, Prof. Joydeep Ghosh for his inspiration and valuable guidance at all stages of work. He has been extremely encouraging and gave insightful inputs towards the culmination of the report. I am also grateful to Dr. Sujay Sanghavi for consenting to be the committee member for the report. I thank my colleagues, Sindhu Vijaya Raghavan and Neeraj Gaur who have worked with me on projects described in the report. I thank the administrators and members of *Intelligent Data Exploration and Analysis (IDEA) Laboratory* at UT Austin for providing me with necessary resources for experimentation. I thank Sreangsu Acharyya and Sanmi Koyejo for their help and suggestions. Finally, I thank the administrative staff of Electrical and Computer Engineering department at UT Austin for their support.

A Survey on using Side Information in Recommendation Systems

Suriya Gunasekar, M.S.E
The University of Texas at Austin, 2012

Supervisor: Joydeep Ghosh

This report presents a survey of the state-of-the-art methods for building recommendation systems. The report mainly concentrates on systems that use the available side information in addition to a fraction of known affinity values such as ratings. Such data is referred to as *Dyadic Data with Covariates (DyadC)*. The sources of side information being considered includes user/item entity attributes, temporal information and social network attributes. Further, two new models for recommendation systems that make use of the available side information within the *collaborative filtering (CF)* framework, are proposed. *Review Quality Aware Collaborative Filtering*, uses external side information, especially review text to evaluate the quality of available ratings. These quality scores are then incorporated into probabilistic matrix factorization (PMF) to develop a *weighted* PMF model for recommendation. The second model, *Mixed Membership Bayesian Affinity Estimation (MMBAE)*, is based on the paradigm of *Simultaneous Decomposition and Prediction (SDaP)*. This model simultaneously learns mixed membership cluster assignments for users and items along with a predictive model for rating prediction within each co-cluster. Experimental evaluation on benchmark datasets are provided for these two models.

Table of Contents

Acknowledgments **v**

Abstract **vi**

List of Tables **iii**

List of Figures **iv**

Chapter 1. Introduction **1**

1.1 Recommendation Systems 4

1.1.1 Content based Recommendation Systems 4

1.1.2 Collaborative filtering based Recommendation Systems 5

1.1.3 Hybrid Recommendation Systems 6

1.2 Notation 7

Chapter 2. Latent Factor Models **9**

2.1 Matrix Factorization based methods 10

2.1.1 Probabilistic Matrix Factorization 11

2.1.2 Bayesian Probabilistic Matrix Factorization 15

2.1.3 Generalized Probabilistic Matrix Factorization 17

2.2 Clustering based methods 19

2.2.1 Collaborative filtering via co-clustering 21

2.2.2 Mixed Membership Stochastic Block models (MMSB), Bayesian co-clustering (BCC) and Residual Bayesian co-clustering (RBC) 23

2.2.3 Mixed Membership Matrix Factorization 26

Chapter 3. Using Side Information in Affinity Estimation **29**

3.1 Using Entity Attributes 31

3.1.1 Regression Based Latent Factor Model 33

3.1.2 fLDA 36

3.1.3 Latent Feature log-linear (LFL) Model . . . 37
3.1.4 Generalized Probabilistic Matrix Factorizations(GPMF) with Side Information . . . 39
3.1.5 Bayesian Matrix Factorization with Side Information (BMFSI) . . . 42
3.1.6 Predictive Discrete Latent Factor Model (PDLF) . . . 44
3.1.7 Simultaneous Co-clustering And Learning (SCOAL) . . . 45
3.2 Using Temporal Dynamics . . . 46
3.2.1 Time Aware SVD (timeSVD) . . . 47
3.2.2 Bayesian Probabilistic Tensor Factorization (BPTF) . . . 48
3.3 Using Network Structure . . . 49
3.3.1 SoRec: Social Recommendation . . . 52
3.3.2 Recommendations using Social Trust Ensemble (STE) . . . 54
3.3.3 Social Matrix Factorization (SocialMF) . . . 55
3.3.4 Kernelized Probability Matrix Factorization (KPMF) . . . 56
3.4 Observation Sensitivity . . . 58

Chapter 4. Review Quality Aware Collaborative Filtering 62
4.1 Model description . . . 64
4.1.1 Stage 1: Quality Score Estimation . . . 65
4.2 Experiments . . . 67
4.3 Results and Discussion . . . 71
4.3.1 Quality Indicators . . . 73

Chapter 5. Mixed Membership Bayesian Affinity Estimation 76
5.1 Model Description . . . 77
5.2 Inference . . . 79
5.2.1 E-Step . . . 80
5.2.2 M-Step . . . 81
5.3 Experiments and Results . . . 82

Chapter 6. Conclusion and Future Work 84
6.1 Future Work . . . 85

Bibliography 87

List of Tables

4.1 Various statistics about the data sets used in experimental evaluation. . . . 68

4.2 Test RMSE for Books and Audio CDs data sets in Stage 2. "LR", "LASSO", and "SVR" refer to the models in which quality scores are predicted using logistic regression, LASSO regression, and support vector regression respectively. 72

5.1 Validation and Test RMSE on MovieLens dataset 83

List of Figures

2.1 (a)Probabilistic Matrix Factorization Model and (b) Bayesian Probabilistic Matrix Factorization . . . 12

2.2 Generalized PMFs(a)Parametric PMF and (b) Mixture PMF . . . 18

2.3 Bayesian Co-Clustering . . . 24

2.4 Mixed Membership Matrix Factorization . . . 27

3.1 A summary relating the latent factor models discussed in the report . . . 30

3.2 Regression Based Latent Factor Model . . . 34

3.3 fLDA . . . 38

3.4 Generalized PMFs with Side Information . . . 42

3.5 Bayesian Matrix Factorization with side information . . . 43

3.6 Social Recommender . . . 54

3.7 Trust Based CF . . . 55

4.1 Graphical model for the two stage approach to recommender systems . . . 64

4.2 Distribution of quality scores from LR-metadata, the best performing model on the Books data set. . . . 73

5.1 Mixed Membership Bayesian Affinity Estimation (MMBAE) . . . 77

Chapter 1

Introduction

The growth in e-commerce industry in recent years has lead to availability of large amounts of data. This report is centered around data mining problems that use data containing affinity relationship between two or more entities. Bi-modal measurements, such as the affinity data described are also known as "dyadic data". One of the popular sources of dyadic data is the item recommendation system, where the pair of entities involved are users and items (movies, music, products, advertisements etc.) and the affinity response is in the form of implicit or explicit preference of a user for an item. Analogously, in social networks, both the entities involved are users and affinities are the strength of the links connecting pairs of users. Other sources of dyadic data include scientific literature, where links could be formed between publications through citations, web pages that form network through web-links, protein interaction networks that connect different proteins by their physical bindings, etc. The task of estimating the unknown affinity values from a subset of known values (and possible external side-information about the dyad) leads to a widely studied class of "affinity estimation" problems in data mining. In particular, recommender systems are deployed for one such task of identifying personalized preferences for users in the system. Such systems can greatly enhance the user experience in e-commerce and effective marketing strategies, such as personalized ranking of search queries, ad targeting etc., can be formulated with the predicted affinities. Affinity estimation is a vast field in itself, this report primarily

discusses affinity estimation in recommender systems with emphasis on systems that use available side-information about the entities in addition to known affinity values. Such data is referred as "Dyadic Data with Covariates (DyadC)". Some of the ideas discussed in this report can be extended to other affinity estimation problems.

Recommender systems have gained prominence owing to the advent of innumerable services available on the internet. Some of the successfully deployed large-scale recommender systems include movie/music recommendation (Pandora[1], Netflix[2]), news recommendation (Yahoo! Inc.[3], Google Inc.[4]) and product recommendation by online retailers like Amazon.com[5] and eBay.com[6]. The announcement of Netflix prize challenge[7] led to aggressive research in this field towards improving the accuracy of existing systems. These systems aid the end user in filtering plethora of choices available to them by providing personalized recommendations. The growing amount of data opens up numerous research avenues for building large scale accurate recommendations. These systems primarily make use of known values of affinities available for a subset of user-item pair. The known affinity values are obtained through either explicit feedback or implicit feedback. Explicit feedback are explicit inputs given by users regarding their interest in products (like ratings). Implicit feedback are indirectly estimated by observing user behavior like purchase history, browsing history, search patterns etc. Apart from the affinity values, many systems also contain side-information associated with the entities which could be useful for user and item profiling.

[1]http://www.pandora.com
[2]https://www.netflix.com
[3]http://www.yahoo.com
[4]http://www.news.google.com
[5]http://www.amazon.com
[6]http://www.ebay.com
[7]http://www.netflixprize.com/

Some of the main challenges associated with present day recommendation systems include high data sparsity, scalability with respect to the growing amount of data, handling *cold start* (a scenario when a user or item has no previous rating available), privacy concerns of users and handling noise in the data including presence of *shillings* (spurious ratings intended to maliciously promote or degrade products), *synonymy* (similar items having different names), etc. Further, the distribution of available observations is highly non-random and a small fraction of entities typically account for a large fraction of data. Moreover, the system is not static and often the factors influencing the predictions and their values change over time. Some of these problems and their existing remedies are discussed in [1].

The estimation of affinity responses can be significantly aided by a variety of available sources of secondary information about the dyadic entities. The side-information includes, user specific features, like age, gender, demographics and network information, item specific information, like product description, performing artists in movies and thematic tags and dyad specific features, like time stamp, users' explicit affinity for items' features, etc. Such information are derived from a variety of sources including entity characteristics (covariates), temporal information, underlying social network and domain knowledge of data generation process (are the elements missing at random or not), etc. Both, the auxiliary attributes and the past responses associated with entities can have a strong bearing on the affinity relationships. For example, users of certain age group tend to enjoy similar genre of movies and music, users who are "friends" in a social network tend to have similar tastes in products, the ratings provided in the latest past might be more reliable for prediction in near future than the ratings provided earlier in time, etc. Efficient use of side-information and past interaction in a single framework allows seamless handling of both "light" and

"heavy" users – while entities with large number of ratings can get accurate predictions through past interactions, on the other hand, previously unseen entities can fall back on the estimates based on the associated side-information. Chapter 3, discusses some of the popular systems for affinity estimation using side-information from various sources of data.

1.1 Recommendation Systems

Traditionally, two broad strategies have been applied while generating recommendations [1, 2]. Content based approaches [3, 4] make use of characteristic features of users and items (such us demographic information and product descriptions), to create entity profiles. The predictions are then generated solely based on these entity profiles. An alternative approach is collaborative filtering (CF) [5–8], based models that make use of past user interactions for future predictions. Both, content based recommender systems and CF systems have limitations. While CF systems do not explicitly incorporate feature information, content-based systems do not incorporate the information in preference similarity across individuals. Hybrid techniques [9–11], combine CF and content based techniques, hoping to avoid the limitations of either approach and thereby improve recommendation performance. These techniques are briefly described below.

1.1.1 Content based Recommendation Systems

Content based filtering methods are based on information about the characteristic features of items and users involved in affinity estimation. The profiles of users and items generated using these features are then used by various algorithms to make affinity prediction for a particular dyad. One example is the work proposed by Balabanovic et. al. [3] which recommends Web pages to users, using similarity of web pages, calculated based on

the 100 most important words. Explicit feedback from a user is usually used to assign weights on the importance of attributes. Another content based web page recommendation system, proposed by Pazzani et. al. [4] uses a naive Bayes classifier to classify web pages using the explicitly available user information. Though such systems have been successfully deployed industrially, often rich external information about users and items may not be available which leads to poor performance of these systems. Another successful paradigm of recommendation systems is the Collaborative Filtering.

1.1.2 Collaborative filtering based Recommendation Systems

The premise behind collaborative filtering (CF) is that users who have agreed in the past tend to agree in the future. CF systems aim to provide predictions about the interests of a user for an item using the previously collected affinity data for similar users and/or items. These systems analyze relationships between users and interdependencies among products to identify new user-item associations. CF approaches have been primarily categorized as follows.

- **Memory Based Systems**

  This approach uses users' past ratings data to compute similarity between users and/or items, and affinities are predicted as a weighted average of affinity values from similar users (user-based recommender systems) or items (item-based recommender systems) [5, 6, 12, 13]. In k-nearest neighbor, item-based systems [6], the similarities between past ratings for different items in the dataset are calculated and affinity of a user i for an item j is estimated as a weighted average of the ratings provider by user i for k items, most similar (based on some similarity measure) to j. Analogously, in user-

based neighborhood methods, user-similarities are used for recommendations. The recommendations thus provided are interpretable and the system is easy to update. However, the disadvantage of such a system is that the entire past affinity data needs to be loaded in memory and an exponential number of pairwise similarly computations are required in the system. Thus, these methods can be expensive in terms of memory and computation for large datasets.

- **Model Based Systems**

  Model based CF algorithms learn user ratings through parametric learning models [7, 14–16] such as classification, regression, clustering and dependency networks. The known affinities are used to train the parameters of the model. These have been used to overcome the shortcomings of memory based CF algorithms as the predictions are based solely on model parameters and training data is no longer is needed in the memory. Among these models, there is a large class of highly successful latent factor models for CF that are discussed in detail in Chapter 2

The collaborative filtering systems are domain independent and they are generally more accurate that content based methods. The major shortcoming of this approach is its inability to predict ratings for users and/or items with no ratings history, this problem is known as cold start.

1.1.3 Hybrid Recommendation Systems

Several recommendation systems use a hybrid approach by combining collaborative and content based methods ([3, 9–11, 17] and references therein). The collaborative and content based recommendations can be combined by either combining the predictions

of separately trained recommendation systems or by constructing a combined model that incorporates both factors. However, these models are generally more complex and computationally intensive compared to either of collaborative or content based methods. Some of these models are briefly described in Section 3.1.

Finally, ensemble methods use multiple models to obtain better predictive performance than those obtained from any of the constituent models [18, 19]. Individual outputs from separate models are typically combined into a unified single output by applying a combination rule. Ensemble learning methods can often outperform single strong learners in practical applications.

1.2 Notation

Unless specified otherwise, the following notation is followed throughout the report. The dyadic data consists of M users indexed by $i \in \{1, 2, \ldots, M\}$ and N items indexed by $j \in \{1, 2, \ldots, N\}$. The user-item rating matrix is represented by $Y \in \mathbb{R}^{M \times N}$ and y_{ij} represents the rating given by user i to item j. The dyad indices of the set of observed ratings is denoted by κ and Y_κ represents the subset of response matrix entries which are observed. In matrix factorization based models, user and item latent factor matrices are represented by $U \in \mathbb{R}^{M \times D}$ and $V \in \mathbb{R}^{N \times D}$ respectively, where D is the number of latent factors in the model. The user and item latent factors for user i and item j are denoted by $\boldsymbol{u_i}$ and $\boldsymbol{v_j}$ respectively.

The bold face variables represent vectors and capitalized alphabets represent matrices. Also, $\mathcal{N}(x|\mu, \sigma^2)$ represents a univariate Gaussian distribution with mean μ and variance σ^2, evaluated at sample point x, and $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \Sigma)$ represents a multivariate Gaussian

distribution with mean $\boldsymbol{\mu}$ and covariance matrix of Σ evaluated at sample point $\boldsymbol{x}$.

Finally, in all the graphical models presented, the circular unshaded nodes represent the latent or hidden variables, the observed variables are shaded circular nodes and the model parameters to be learned are rectangular nodes.

The rest of the report is organized as follows. Chapter 2 gives an overview of the existing latent factor based CF models which solely use the affinity values. Chapter 3 discusses the systems for affinity estimation using side information from various sources of data. Chapter 4 describes the *Review Quality Aware Collaborative Filtering* model, that uses text reviews towards building quality aware models and experiments are provided to support the efficacy of the model. Chapter 5 describes, the *Mixed Membership Bayesian Affinity Estimation* model developed for affinity estimation using side-information. Finally, Chapter 6 concludes the report with some ideas for future work.

Chapter 2

Latent Factor Models

Recommendation systems based on latent factor models are among the widely used methods for collaborative filtering. These are model based techniques, where the observed data is explained using hidden or latent random variables. The idea behind latent factor models is that the attitudes or preferences of a user are determined by a small number of unobserved factors. The latent factor models in turn can be categorized as matrix factorization based models and clustering based models. The former approximates the response matrix as a product of low rank matrices. In the later approach, the user and/or items are typically clustered into homogeneous groups based on their associated latent factors, and predictions are dictated by their cluster memberships.

The latent factors and model parameters in these models are learned by maximizing the posterior distribution of the latent variables conditioned on the observed data. For Bayesian models, model hyperparameters are learned by maximizing the observed likelihood which is obtained by marginalizing over the latent variables. In most of these methods, it is difficult to evaluate in exact form of the marginals and hence approximate methods are used for inference. Two most popular approximate inference algorithms are *variational methods* [20] and *sampling methods* [21]. Variational methods lower bound the posterior distribution of latent variables conditioned on the observed variables using Jensen's Inequality [22], and finds the member of a restricted family of posteriors that best approximates the true

posterior in terms of KL (KullbackLeibler) divergence. On the other hand, sampling based methods approximate the posterior using the mean of samples drawn through Monte Carlo simulations.

2.1 Matrix Factorization based methods

In linear factor models, each users' preference for an item is modeled as a linear combination of small number of item latent factors with the user specific affinity for those factors (user factors). This idea leads to formulation matrix factorization based CF models. These models approximate the user-item rating matrix, Y, as a dot product of two low-rank matrices - the user-factor matrix, $U \in \mathbb{R}^{M \times D}$ and the item-factor matrix, $V \in \mathbb{R}^{N \times D}$, $Y = UV^T$. Matrix factorization based models are among the most scalable algorithms for CF.

The early work in this direction include [23] and [16]. Both these models use low rank approximations as a preprocessing step. The former performs a low rank factorization on the fully observed subset of columns of the matrix and unobserved item ratings are computed by projecting the items into the space spanned by the fully observed columns. The later work performs a singular value decomposition (SVD) on sparse binary observation matrix and then learns preferences through neural networks. In [24], an EM based algorithm to solve weighted low-rank approximation of matrices was applied to collaborative filtering by weighing the observed entries by 1 and the unobserved ones by 0.

In SVD (Singular Value Decomposition) based methods, low-rank approximation, X, of the complete matrix, Y, is obtained by minimizing the sum-squared error over the observed entries with a constraint on the rank of X. A convex proxy for the above problem

was proposed by [25]. The authors formulate a semi-definite program (SDP), where the rank constraint on X is replaced by a constraint on the trace norm, $\|X\|_\Sigma$. A scalable version of the above problem was formulated by Rennie et. al [26]. They propose an SVD based algorithm that learns the matrices U and V of given rank such that $\|Y - UV^T\|_F^2$ is minimized over the observed entries of the target matrix Y. The regularization is provided by constraining the Frobenius norm of the latent factors which was shown to be equivalent to constraining the trace norm of UV^T.

$$\begin{aligned} U, V &= \underset{U,V}{argmin} \quad \sum_{(ij)\in\kappa} \|y_{ij} - \boldsymbol{u_i}^T \boldsymbol{v_j}\|_2^2 \; + \; \frac{C}{2}(\|U\|_F^2 \; + \; \|V\|_F^2) \\ &= \underset{U,V}{argmin} \quad \sum_{(ij)\in\kappa} \|y_{ij} - \boldsymbol{u_i}^T \boldsymbol{v_j}\|_2^2 \; + \; C\|UV^T\|_\Sigma \end{aligned} \tag{2.1}$$

where the $\|X\|_\Sigma$ and $\|X\|_F = (\sum_{ij} |X_{ij}|^2)^{\frac{1}{2}}$ are the trace norm and the Frobenius norm of a matrix X respectively. Gradient descent based methods were used in the later work leading to a more scalable implementation.

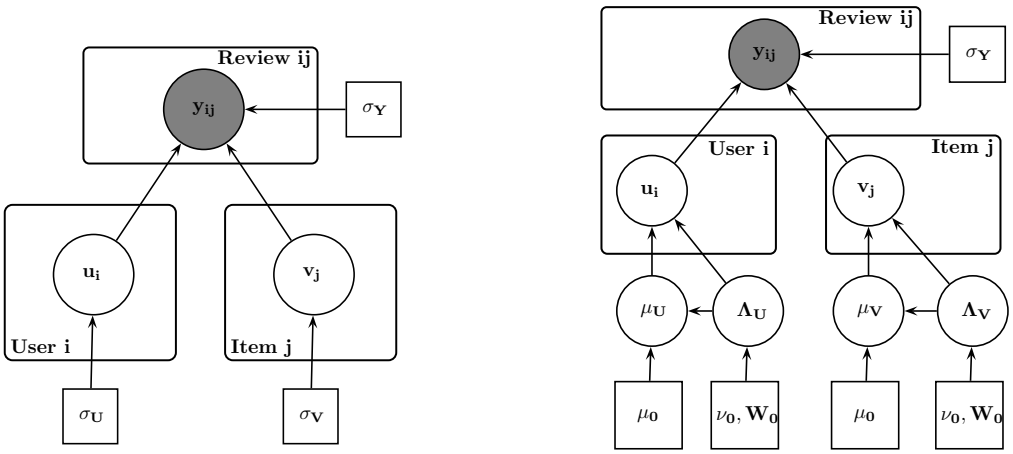
An alternate objective to the above discussed SVD model was proposed by Koren et. al. [27]. To explain the effects of user and item biases, the authors proposed an interaction independent bias term. The ratings in the new model were explained by both the bias term and the interaction term.

$$y_{ij} = \mu + \alpha_i + \beta_j + \boldsymbol{u_i}^T \boldsymbol{v_j} \tag{2.2}$$

where, μ is the global average of the ratings, α_i and β_j are the user and item bias respectively. The $\{\alpha_i\}$ and $\{\beta_j\}$ are learned as latent factors of the model.

2.1.1 Probabilistic Matrix Factorization

Probabilistic Matrix factorization proposed by Salakhutdinov [8] was a base model for many approaches developed later. The computational cost for the model scales linearly



(a) PMF (b) BPMF

Figure 2.1: (a)Probabilistic Matrix Factorization Model and (b) Bayesian Probabilistic Matrix Factorization

with the number of observations and was the first of its kind that scaled to large datasets. The graphical model representing the model is shown in Figure 2.1a.

The ratings are modeled as a linear combination of user and item factors perturbed by a Gaussian observation noise. The choice of Gaussian noise is motivated by the use of RMSE (Root Mean Squared Error) as evaluation criterion. Further, a zero mean Gaussian prior is placed on the user and item factors. Thus, the conditional distribution over observed ratings is given by:

$$P(Y|U,V,\sigma_Y^2) \;=\; \prod_{ij}[\mathcal{N}(y_{ij}|\boldsymbol{u_i}^T\boldsymbol{v_j},\sigma_Y^2)]^{I_{ij}} \tag{2.3}$$

where I_{ij} is the indicator variable for observed elements of the matrix, i.e. $I_{ij} = 1$ if Y_{ij} is observed, else $I_{ij} = 0$. Also,

$$\begin{aligned} P(U|\sigma_U^2) &= \prod_{i=1}^{M}\mathcal{N}(0,\sigma_U^2) \\ P(V|\sigma_V^2) &= \prod_{j=1}^{N}\mathcal{N}(0,\sigma_V^2) \end{aligned} \tag{2.4}$$

where, σ_U^2, σ_V^2 and σ_Y^2 are the variances associated with the distributions of user and item latent factors and observed ratings respectively.

Maximizing the log-posterior of the latent variables conditioned on the observed ratings and the model parameters is equivalent to minimizing the following objective of sum squared error with quadratic regularization terms:

$$U, V = \underset{U,V}{argmin} \sum_{i=1}^{M} \sum_{j=1}^{N} I_{ij}(y_{ij} - \boldsymbol{u_i}^T \boldsymbol{v_j})^2 \ + \ \lambda_U \|U\|_F^2 \ + \ \lambda_V \|V\|_F^2 \tag{2.5}$$

where $\lambda_U = \sigma_Y^2/\sigma_U^2$ and $\lambda_V = \sigma_Y^2/\sigma_V^2$ are the regularization parameters. A gradient descent that has a time complexity linear in the number of observations, is performed to obtain the local minimum of the above objective.

The model complexity in PMF is controlled by the hyperparameters λ_U and λ_V. A traditional method for tuning these hyperparameters involves evaluating the performance of models trained with different values of regularization parameters on a held out validation set. This process is highly computation intensive. To overcome this problem, the authors propose including adaptive priors on the model parameters to automatically control model complexity. The idea was motivated from the method proposed by [28], originally applied to neural networks. The automatic complexity control is achieved by introducing priors on the hyperparameters of the latent factors. The model is then learned by maximizing the log posterior over both latent factors and hyperparameters. In the special case of using spherical Gaussian priors for user and movie feature vectors leads to the standard form of PMF with λ_V and λ_U chosen automatically. This approach further allows use of regularization methods that are more sophisticated than penalization of the squared norm. The log posterior to be

maximized is of the form:

$$L(U, V, \Theta_U, \Theta_V) \;=\; \ln P(Y|U,V) + \ln P(U|\Theta_U) + \ln P(V|\Theta_V) + \ln P(\Theta_U) + \ln P(\Theta_V) \quad (2.6)$$

where, Θ_U and Θ_V are the hyperparameters defining the prior distributions on U and V respectively. The model is learned by alternating between optimizing the hyperparameters and updating the feature vectors using steepest ascent.

Finally, the authors propose a constrained version of PMF that was shown to perform better for users with very few ratings. This is based on the assumption that users who have rated similar set of movies tend to have similar preferences. The user latent factor is constrained as follows:

$$\boldsymbol{u_i} \;=\; \alpha_i + \frac{\sum_{j=i}^{N} I_{ij} \boldsymbol{w_j}}{\sum_{j=1}^{N} I_{ij}} \quad (2.7)$$

where $W \in \mathbb{R}^{N \times D}$ is the latent similarity matrix with prior $P(W|\sigma_W) = \prod_{j=1}^{N} \mathcal{N}(0, \sigma_W^2 I)$ and α_i is the user bias. The jth row of matrix W captures the effect that a user rating an item j has on the user factors. Thus, users who have rated similar movies are constrained to have similar latent factors. The model is now learned over V, W and α.

The experimental results showed improvement over the traditional SVD models. Moreover, PMF with adaptive priors showed better performance than the fixed hyperparameter model and the constrained PMF model performs much better and converges considerably faster than the unconstrained PMF models. The main contributions of the paper include formulation of a scalable probabilistic model for matrix factorization with provisions for automatic complexity control and its extension to make accurate predictions for users with very few ratings. Most of the preceding work on collaborative filtering tend to ignore users with very few ratings from the dataset as those models perform poorly for such users.

2.1.2 Bayesian Probabilistic Matrix Factorization

PMF model described above showed superior performance on many experimental datasets compared to the previously existing collaborative filtering techniques. However, as learning the posterior distribution over latent factors $P(U, V|Y_\kappa)$ is intractable, the MAP (Maximum a posteriori) estimates for U and V are used. As with all single point estimates, unless the regularization parameters are tuned carefully, this approach is prone to over-fitting. Alternatively, automatic complexity control suggested in [28] introduces priors for the hyperparameters and maximizes the log-posterior of the model over both parameters and hyperparameters. However, this approach lacks theoretical grounding and the point estimate thus obtained can again be skewed in sparse datasets.

Most CF models are intended for the task of predicting unobserved entries of Y rather than estimating the user and item factors. The Bayesian approach to PMF involves integrating out the model parameters to directly predict unobserved entries in the ratings matrix. In the work by Ruslan et. al. [29], a fully Bayesian treatment of PMF (BPMF) model was proposed. The authors provide tractable approximate inference to the model using Markov chain Monte Carlo (MCMC) simulations on large scale sparse datasets. The graphical model representing Bayesian PMF is shown in Figure 2.1b.

As in PMF, the likelihood of observed ratings is given by Equation 2.3. However, the priors on the user and item factors are assumed to be Gaussian distributed as follows:

$$\begin{aligned} P(U|\boldsymbol{\mu_U}, \Lambda_U) &= \prod_{i=1}^{M} \mathcal{N}(\boldsymbol{u_i}|\boldsymbol{\mu_U}, \Lambda_U^{-1}) \\ P(V|\boldsymbol{\mu_V}, \Lambda_V) &= \prod_{j=1}^{N} \mathcal{N}(\boldsymbol{v_i}|\boldsymbol{\mu_V}, \Lambda_V^{-1}) \end{aligned} \tag{2.8}$$

where $\Lambda \in \mathbb{R}^{D \times D}$ is the inverse covariance matrix and $\Theta_U = \{\boldsymbol{\mu_U}, \Lambda_U\}$ and $\Theta_V = \{\boldsymbol{\mu_V}, \Lambda_V\}$ are the user and item hyperparameters respectively. Further, the user and item hyperparameters are given Gaussian-Wishart conjugate priors:

$$P(\Theta_U|\boldsymbol{\mu_0}, \beta_0, \nu_0, W_0) \;=\; \mathcal{N}(\boldsymbol{\mu_U}|\boldsymbol{\mu_0}, (\beta_0\Lambda_U)^{-1})\mathcal{W}(\Lambda_U|W_0, \nu_0) \tag{2.9}$$

$$P(\Theta_V|\boldsymbol{\mu_0}, \beta_0, \nu_0, W_0) \;=\; \mathcal{N}(\boldsymbol{\mu_V}|\boldsymbol{\mu_0}, (\beta_0\Lambda_V)^{-1})\mathcal{W}(\Lambda_V|W_0, \nu_0) \tag{2.10}$$

where $\mathcal{W}(\Lambda|W, \nu) \propto |\Lambda|^{\frac{\nu_0 - D - 1}{2}} exp(-\frac{1}{2}Tr(W_0^{-1}\Lambda))$ is a $D \times D$ Wishart distribution with ν degrees of freedom and scale matrix W. The model parameters are denoted by $\Theta_0 = \{\boldsymbol{\mu_0}, \beta_0, \nu_0, W_0\}$.

The predictive distribution for an unobserved element of the ratings matrix is given by:

$$P(y_{ij}|Y_\kappa, \Theta_0, \sigma_Y^2) = \int P(y_{ij}|\boldsymbol{u_i}, \boldsymbol{v_j}, \sigma_Y^2)P(\boldsymbol{u_i}|\Theta_U)P(\boldsymbol{v_j}|\Theta_V)P(\Theta_U, \Theta_V|\Theta_0)\; dU dV d\Theta_U d\Theta_V \tag{2.11}$$

The authors use Markov Chain Monte Carlo (MCMC) simulations for inference. The conjugate priors on the hyperparameters lead to Gaussian distributions of the posteriors of individual factors, which are easy to sample from. Thus, Gibbs sampling was used to perform MCMC simulations. This leads to a trivially parallelizable inference algorithm that is scalable to large datasets. The algorithm is described in the Algorithm 1.

The results show that BPMF considerably improves predictive accuracy compared to standard PMF models. An added advantage of BPMF is that the predictive distribution quantifies the confidence in predictions which can be taken into account while making recommendations using the model. On the other hand, the Gibbs Sampling step for inference

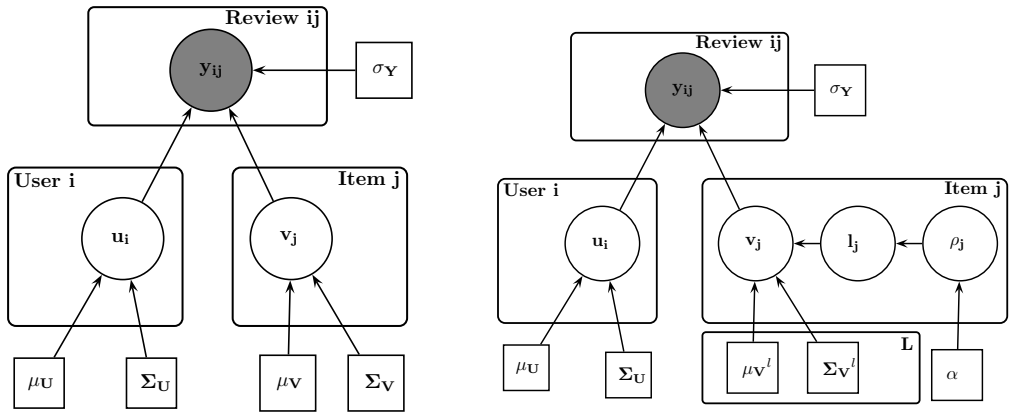
Algorithm 1 Gibbs Sampling Routine for BPMF

Initialize $U^{(0)}$,$V^{(0)}$
for $k = 1, 2, \ldots, K$ **do**
  Sample $\Theta_U^{(k)} \sim P(\Theta_U^{(k)}|U^{(k-1)}, \Theta_0)$
  Sample $\Theta_V^{(k)} \sim P(\Theta_V^{(k)}|V^{(k-1)}, \Theta_0)$
  for $i = 1, 2, \ldots, M$ **do**
    Sample in parallel $\boldsymbol{u_i^{(k)}} \sim P(\boldsymbol{u_i^{(k)}}|Y_\kappa, V^{(k-1)}, \Theta_U^{(k)})$
  end for
  for $j = 1, 2, \ldots, N$ **do**
    Sample in parallel $\boldsymbol{v_j^{(k)}} \sim P(\boldsymbol{v_j^{(k)}}|Y_\kappa, U^{(k)}, \Theta_V^{(k)})$
  end for
end for
$P(y_{ij}^*|Y_\kappa, \Theta_0) \approx \frac{1}{K}\sum_{k=1}^{K} P(y_{ij}^*|U^{(k)}, V^{(k)}, \sigma_Y^2)$

is computationally intensive. The trade off between the computation and accuracy depends in the dataset and the end application involved.

2.1.3 Generalized Probabilistic Matrix Factorization

A set for extensions to PMF termed as "Generalized PMFs (GPMFs)", was proposed by Shan et. al. [30]. The first two models proposed in the paper are discussed here. The other models use side information and are discussed in Chapter 3. In "Parametric PMF (PPMF)" model the independence assumption on latent factors (as suggested by PMF model using diagonal covariance matrix) is removed by allowing for non-diagonal covariance matrices, $\Sigma_U \in \mathbb{R}^{D\times D}$ and $\Sigma_V \in \mathbb{R}^{D\times D}$ for user and item factors respectively. Also, the model allows for non zero means, $\boldsymbol{\mu_U} \in \mathbb{R}^D$ and $\boldsymbol{\mu_V} \in \mathbb{R}^D$, for the latent factors. The PPMF model learns the parameters $\Theta = \{\boldsymbol{\mu_U}, \boldsymbol{\mu_V}, \Sigma_U, \Sigma_V, \sigma_Y\}$, by maximizing the posterior of the latent variables. The idea is an application of automatic complexity control extension suggested in PMF. A variational EM approximation is proposed for the inference. *Mean field approximation* [31] is used for inference and a MAP estimate of the unobserved



(a) PPMF (b) MPMF

Figure 2.2: Generalized PMFs(a)Parametric PMF and (b) Mixture PMF

ratings is used for prediction.

The second model proposed in the paper is the "Mixture Probabilistic Matrix Factorization (MPMF)" model. In this model, the item factors are derived from a *mixture of Gaussian* prior. The item factors $\boldsymbol{v_j}$ are sampled from a weighted mixture of L Gaussian distributions with mean $[\boldsymbol{\mu_V^l}]_{l=1}^L$ and covariance matrices $[\Sigma_V^l]_{l=1}^L$ respectively. The generative process for the model is described in Model 2. In principle, the mixture of Gaussian models can also be extended to the user factors. The model parameters are the ML (maximum likelihood) estimates which are learned through variational methods.

The graphical models representing PPMF and MPMF are shown in Figures 2.2a and 2.2b respectively. On one hand, the experiments show improvements over vanilla PMF. On the other hand, for small and moderate sized data, the additional parameters involved in these models could potentially overfit the data.

Model 2 Generative Process for MPMF

for $i = 1, 2, \ldots, M$ **do**
  Sample $\boldsymbol{u_i} \sim \mathcal{N}(\boldsymbol{\mu_U}, \Sigma_U)$
end for
for $j = 1, 2, \ldots, N$ **do**
  Sample $\boldsymbol{\rho_j} \sim Dir(\boldsymbol{\alpha})$
  Sample $l_j \sim Mult(\boldsymbol{\rho_j})$
  Sample $\boldsymbol{v_j} \sim \mathcal{N}(\boldsymbol{\mu_V^{l_j}}, \Sigma_V^{l_j})$
end for
for $(i, j) \in (M \times N)$ **do**
  Sample $y_{ij} \sim \mathcal{N}(y_j | \boldsymbol{u_i}^T \boldsymbol{v_j}, \sigma_Y^2)$
end for

2.2 Clustering based methods

In the context of recommendation systems, data is generated from a heterogeneous mixture of users and items. In such cases, a single model could be inadequate in explaining the dynamics of space of large number of user preferences with limited data. One of the most common approaches in such scenario is to cluster the sample points into homogeneous subgroups and then learn predictive models for each subgroup. This two-step procedure usually results in simpler, more interpretable models. A class of clustering based latent factor models are developed based on this idea. The users and/or items are given hard or soft cluster assignments based on similarity of latent variables associated with the entities and the affinities are dictated by cluster memberships. The most successful of these approaches are based on co-clustering which is simultaneous clustering of rows and columns of dyadic data.

Some of the early work in this direction involved clustering of user profiles. This includes *multinomial mixture* models, *aspect models* [13], and *User Rating Profile (URP)* models [32]. These approaches primarily model the user profiles (also referred as user attitudes) through a set of latent variables representing predefined "typical" interest patterns.

The ratings of the users are explained based on their profiles. In multinomial mixture model [13], each user is associated with a latent variable z_i, which explain the set of ratings provided by the user, $\{y_{i(.)}\}$. The ratings are conditionally independent of the user, given z_i. All the users have the same prior distribution over user attitudes, z_i. A variant of multinomial mixture model, the aspect model (described in the same work), allows each user to have different prior distributions over user attitudes. With the increased complexity of the later model, the inference is more complicated and the number of parameters in the model grows linearly with the number of users in the data set. These models are highly restrictive as a single latent variable is meant to explain all the ratings provided by the user. In URP model [32], each user profile is modeled as a mixture of user attitudes, and the mixing proportions are distributed according to a Dirichlet random variable. The rating for each item is generated by selecting a user attitude for the item, and then selecting a rating according to the preference pattern associated with that attitude. The mixture model allows more flexibility for modeling the dynamics in user rating pattern. However, inference becomes intractable and variational approximation methods are used. In a later work [33], *Multiple Multiplicative Factor (MMF)* models were introduced. In these models, all the predictions of a user are explained by a set of K hidden variables. For computational simplicity, the factors are assumed to be independently generated. The latent factor models described so far, preceded the advent of matrix factorization based models. Though they were shown to perform better than the traditional neighborhood based models, the matrix factorization based models significantly outperformed these clustering based models.

Further in this section, some of the recent clustering based models that showed comparable or better performance to matrix factorization models are discussed. Most of

these models are based on co-clustering. Co-clustering or bi-clustering [34] is a clustering paradigm for dyadic data, where the rows and columns are simultaneously clustered leading to formation of contiguous blocks of clusters. These techniques incorporate row clustering information into column clustering and vice versa, to converge at an efficient cluster assignment. This process can be thought of as alternating regularization of row and column clusters with the current estimates of column and row clusters respectively. Such techniques were shown to yield better quality clusters on each side, especially in the presence of sparse data. Such approaches are widely used in text mining, bioinformatics and matrix completion, [35–37]. Various co-clustering algorithms have been proposed in the context of matrix approximations. In the work by Banerjee et. al. [37], and references therein, the authors develop a partitional co-clustering formulation for approximating dyadic matrix with the objective of minimizing the approximation error. In Bregman co-clustering framework described in [37], the approximation error can take any form of a large class of loss functions called Bregman divergences [38] which also includes square loss commonly used in recommendation systems. The idea is to view co-clustering as a lossy compression problem, where the task is to approximate a given matrix using a fixed number of row and column clusters. The cluster assignments are obtained by minimizing the distortion or loss in information.

In the following sub-sections, unless specified otherwise, it is assumed that there are K row clusters and L column clusters.

2.2.1 Collaborative filtering via co-clustering

A dynamic CF model supporting cold start users and items was proposed in [39]. The task is to find the user and item cluster mappings $\rho : \{1, 2, \ldots, M\} \rightarrow \{1, 2, \ldots, K\}$ and

$\gamma : \{1, 2, \ldots, N\} \rightarrow \{1, 2, \ldots, L\}$ such that the estimate of y_{ij} with $\rho(i) = k$ and $\gamma(j) = l$ is given by Equation 2.12, best approximates the observed data.

$$\hat{y}_{ij} = \mu_{kl} + (\mu_u^{(i)} - \mu_u^{(k)}) + (\mu_v^{(j)} - \mu_v^{(l)}) \tag{2.12}$$

where μ_{kl} is the average of ratings in the co-cluster kl, $\mu_u^{(i)}$ and $\mu_v^{(j)}$ are the average of observed ratings of user i and item j respectively in the ratings matrix, and $\mu_u^{(k)}$ and $\mu_v^{(l)}$ are the average of observed ratings of all users in row cluster k and of all items of column cluster l respectively.

The problem is thus formulated as:

$$\rho^*, \gamma^* = \underset{\rho,\gamma}{argmin} \sum_{i=1}^{M} \sum_{j=1}^{N} [(y_{ij} - \hat{y}_{ij})^2]^{I_{ij}} \tag{2.13}$$

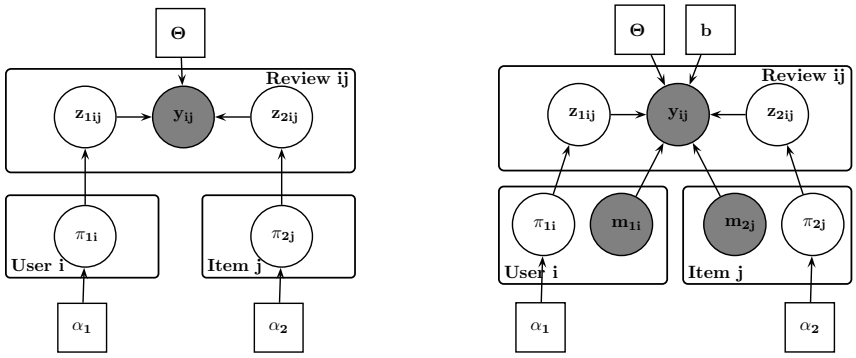
where $\hat{y}_{ij}$ is given by Equation 2.12. The optimization is performed through Bregman co-clustering algorithm described in [37]. This is done by alternatively updating the row and column cluster assignments, $\rho(i)\ \forall i$ and $\gamma(j)\ \forall j$, in a serial manner until convergence. The updates are performed by minimizing the above mentioned cost function. For an existing user and item pair, the prediction is given by Equation 2.12. When one of the user (or item) of the dyad is new, the prediction is just item (or user) average obtained by ignoring user (or item) bias term in the prediction equation. The global average of the co-cluster is returned when both user and item are new. Further, an incremental update of the various cluster and entity means used in prediction can be easily performed to generate a dynamic model.

The model provides comparable performance to SVD based algorithms with reduced computation and ease of update.

2.2.2 Mixed Membership Stochastic Block models (MMSB), Bayesian co-clustering (BCC) and Residual Bayesian co-clustering (RBC)

In co-clustering algorithms considered so far, there is a hard partitioning of users and items into row and column clusters. Essentially, each user (and item) has a single latent role assigned. In many real life data, this might not be the case, a user can exhibit multiple interest patterns at various times. Owing to the inherent mixed membership nature of human users', much of recommendation systems can benefit from mixed membership of row and column entities. This motivates the idea of mixed membership for the users and items in the row and column clusters respectively. The mixed membership relaxation of co-cluster memberships was successfully used in the development of *Mixed Membership Stochastic Block Models* for dyadic data by Airoldi et. al. [40]. Standard mixed membership models like LDA [41], cannot be directly used here as the assumption that the entities are conditionally independent given the cluster membership no longer holds. Instead for dyadic data, it is desirable that the interaction between entities of different clusters be governed by the interaction between the clusters.

The model was originally proposed to handle binary responses between pairs of a single type of entity, like link prediction in social networks. This does not directly apply to the recommendation system setting. However, it is briefly described here as it forms the basis for later developed CF algorithms, [30, 42]. In the generative process, each entity p samples a cluster-membership probabilities, $\boldsymbol{\pi_p} \in \mathbb{R}^K$ from a Dirichlet distribution parameterized by $\boldsymbol{\alpha} \in \mathbb{R}^K$. For each observed link (p, q), a row cluster $z_{p\to q}$ and a column cluster $z_{q\to p}$ are sampled from $\boldsymbol{\pi_p}$ and $\boldsymbol{\pi_q}$ respectively, and the binary response is sampled from a Bernoulli distribution associated with the cluster $z_{p\to q}z_{q\to p}$.



(a) BCC (b) RBC

Figure 2.3: Bayesian Co-Clustering

Shan et. al. proposed mixed membership models, *Bayesian Co-clustering (BCC)* [43] and *Residual Bayesian Co-clustering (RBC)* [42], for recommendation system framework. These models can handle various types of response values that can be generated through a member of *exponential family*[1] of distributions. The generative process is described in Model 3 and the graphical models is given in Figure 2.3a.

Model 3 Generative Process for BCC

for $i = 1, 2, \ldots, M$ **do**
  Sample $\boldsymbol{\pi_{1i}} \sim Dir(\boldsymbol{\alpha_1})$ where $\boldsymbol{\pi_{1i}}, \boldsymbol{\alpha_1} \in \mathbb{R}^K$
end for
for $j = 1, 2, \ldots, N$ **do**
  Sample $\boldsymbol{\pi_{2j}} \sim Dir(\boldsymbol{\alpha_2})$ where $\boldsymbol{\pi_{2j}}, \boldsymbol{\alpha_2} \in \mathbb{R}^L$
end for
for $(i, j) \in (M \times N)$ **do**
  Sample $z_{1ij} = Mult(\boldsymbol{\pi_{1i}})$
  Sample $z_{2ij} = Mult(\boldsymbol{\pi_{2j}})$
  Sample $y_{ij} \sim P_\psi(y_{ij}|\theta_{z_{1ij}, z_{2ij}})$
end for

The matrix with $\boldsymbol{\pi_{1i}}$ for $i \in 1, 2, \ldots, M$ is denoted by $\Pi_1 \in \mathbb{R}^{M \times K}$ and similarly,

[1] A family of probability distribution functions parameterized by θ which are of the form: $p_\psi(x|\theta) = p_0(x)\exp(\langle x, \theta\rangle) - \psi(\theta))$

$\Pi_2 \in \mathbb{R}^{N\times L} = \{[\boldsymbol{\pi_{2j}}]_{j=1}^{N}\}$. The cluster assignment matrices $Z_1 \in \mathbb{R}^{|\kappa|\times K}$ and $Z_2 \in \mathbb{R}^{|\kappa|\times L}$ are defined as the matrices which denote one-hot cluster memberships of users and items respectively for each observed response. Finally, the exponential family parameters for each cluster θ_{kl} are combined into $\Theta = \{[\theta_{kl}]_{(k,l)\in(K\times L)}\}$. The joint likelihood of observed and hidden latent variables is given by:

$$\begin{aligned} P(Y,\Pi_1,\Pi_2,Z_1,Z_2|\boldsymbol{\alpha_1},\boldsymbol{\alpha_2},\Theta) = &\prod_{i=1}^{M} P(\boldsymbol{\pi_{1i}}|\boldsymbol{\alpha_1}) \prod_{j=1}^{N} P(\boldsymbol{\pi_{2j}}|\boldsymbol{\alpha_2}) \\ &\prod_{i=1}^{M}\prod_{j=1}^{N} [P(z_{1ij}|\boldsymbol{\pi_{1i}})P(z_{2ij}|\boldsymbol{\pi_{2j}})P_{\psi}(y_{ij}|\theta_{z_{1ij},z_{2ij}})]^{I_{ij}} \end{aligned} \tag{2.14}$$

The observed likelihood, $P(Y|\alpha_1,\alpha_2,\Theta)$ is obtained by marginalizing over the latent variables. However, it is intractable to evaluate the integral and hence mean field variational methods are used for inference. In mean field variational methods, posterior of latent variable distribution is approximated by a fully factorized form in terms of variational parameters. An EM algorithm is proposed for parameter estimation. In the E-Step of every iteration, the variational distribution closest (in KL divergence measure) to true posterior is obtained by fixing the model parameters. In the M-Step, the model parameters are maximized with respect to the approximate posterior distribution. The prediction on the test set is then estimated by taking the expected value of the response with respect to the cluster assignments.

A variation of BCC using collapsed Gibbs and collapsed variational inference was given in *Latent Dirichlet Bayesian co-clustering* [44]. The authors provide collapsed inference algorithms by marginalizing out Π_1 and Π_2 from the posterior distribution. The new inference showed slight improvement in performance compared to the variational inference proposed in BCC.

As an extension to BCC, Residual Bayesian Clustering(RBC) model was proposed [42]. The model is described in Figure 2.3b. In this model, the co-clustering is inferred from the residual matrix obtained by subtracting a b fraction of the user and item bias, m_{1i} and m_{2j} respectively. The distribution of the response value y_{ij} for a dyad belonging to the co-cluster kl is now given by $p_\psi(y_{ij} - b(m_{1i} + m_{2j})|\theta_{kl})$. A parallelizable EM algorithm for variational inference is proposed in the paper.

2.2.3 Mixed Membership Matrix Factorization

The matrix factorization based models inherently exhibit a static behavior for generation of the ratings, wherein, the entity factors remain constant without regard for the context. On the other hand, mixed membership models introduce context dependency by allowing different choices of models for each rating. However, the relatively poor predictive performance of these models compared to their matrix factorization counterparts suggests that these models have limited expressive power. Mackey et. al. propsed a *Mixed Membership Matrix Factorization* model [45], which tries to combine the discrete mixed membership modeling (that are limited in the expressive power of the prediction model) with continuous latent factor model (that lack context specific dynamics).

In this model, the response value for a dyad is influenced by two components. First is the static component derived as an inner product of user and item latent factors (generated from a BPMF model). The second component is the dynamic mixed membership effect. To generate the mixed membership effect, a cluster membership vectors, $\boldsymbol{\pi_{1i}}$, $\boldsymbol{\pi_{2j}}$ are first sampled by each user, i and item, j (similar to BCC). For each response, a co-cluster, $\{z_{1ij} = k, z_{2ij} = l\}$, is chosen and the component influencing the response is a cluster specific

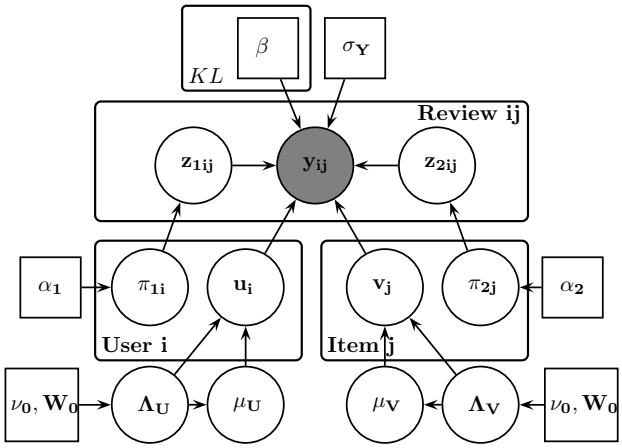


Figure 2.4: Mixed Membership Matrix Factorization

bias, β_{ijkl}, is given by Equation 2.12. The generative process is graphically represented in Figure 2.4.

The response is generated as follows:

$$P(y_{ij}|\boldsymbol{u_i}, \boldsymbol{v_j}, z_{1ij}, z_{2ij}, \beta_{ijkl}, \sigma_Y^2) = \mathcal{N}(y_{ij}|(\beta_{ijz_{1ij}z_{2ij}} + \boldsymbol{u_i}^T \boldsymbol{v_j}), \sigma_Y^2) \tag{2.15}$$

The user and item latent factors are given priors according to Equations 2.8, 2.9 and β_{ijkl} is given by Equation 2.12. The inference is performed through Gibbs Sampling. The model showed significant improvement over the BPMF model, while partially explaining the affinities through the mixed membership model.

In summary, there is a large literature of work using latent factor models for collaborative filtering. This chapter discussed a few popular methods. A unified view of these models using a small set of modeling choices was studied by Singh et. al. [46]. They further propose a generalized alternating projections technique for matrix factorization with a large class of Bregman [38] loss functions. Scalable gradient descent update

algorithms for implementing MF were given in [47]. Some of the other latent factor models include *Bayesian Clustered Tensor Factorization (BCTF)*, [48] and *Multi-Hierarchical Dirichlet Process (Multi-HDP)*, [49]

All the models described in this chapter are pure CF algorithms that use just the past ratings matrix to make predictions for missing entries. Their predictive power for a user (or item) is limited by the number of ratings available for that user (or item). Thus, for cold start cases, these models default to global average that has limited expressive power. Alternate approaches to overcome these problems without loss of accuracy are discussed in the next chapter.

Chapter 3

Using Side Information in Affinity Estimation

As described in Chapter 1, two approaches for recommendation systems are popular. Content based systems build a predictive model entirely based on user and item side-information. Though such a method can handle cold start scenarios, it does not utilize past interaction data in affinity estimation and such systems were shown to be outperformed by CF methods on benchmark datasets, in terms of predictive accuracy. Although effective for warm start, CF systems fail to address cold start.

This chapter focuses on discussing systems that exploit various forms of available side-information for improved performance. The most common form of side information is the user, item and dyad explicit attributes like age, gender, location of the user, bag-of-words description of items etc. Secondly, information relating to the temporal dynamics of the system are shown to give considerable improvements over base models. Thirdly, the underlying social network of the users and other forms of network information can provide useful insights into user and item features. The Figure 3.1, summarizes various models described in this chapter and their relationship to the models in the previous chapter.

The following additional notation is used in this chapter. The number of user, item and cell features are D_1, D_2 and D_0 respectively. The external features associated with the users, items and dyads are given by the matrices $X_1 \in \mathbb{R}^{M \times D_1}$, $X_2 \in \mathbb{R}^{N \times D_2}$ and $X_0 \in \mathbb{R}^{MN \times D_0}$ respectively. The rows of these matrices $\mathbf{x_{1i}}$, $\mathbf{x_{2j}}$ and $\mathbf{x_{0ij}}$ represent the

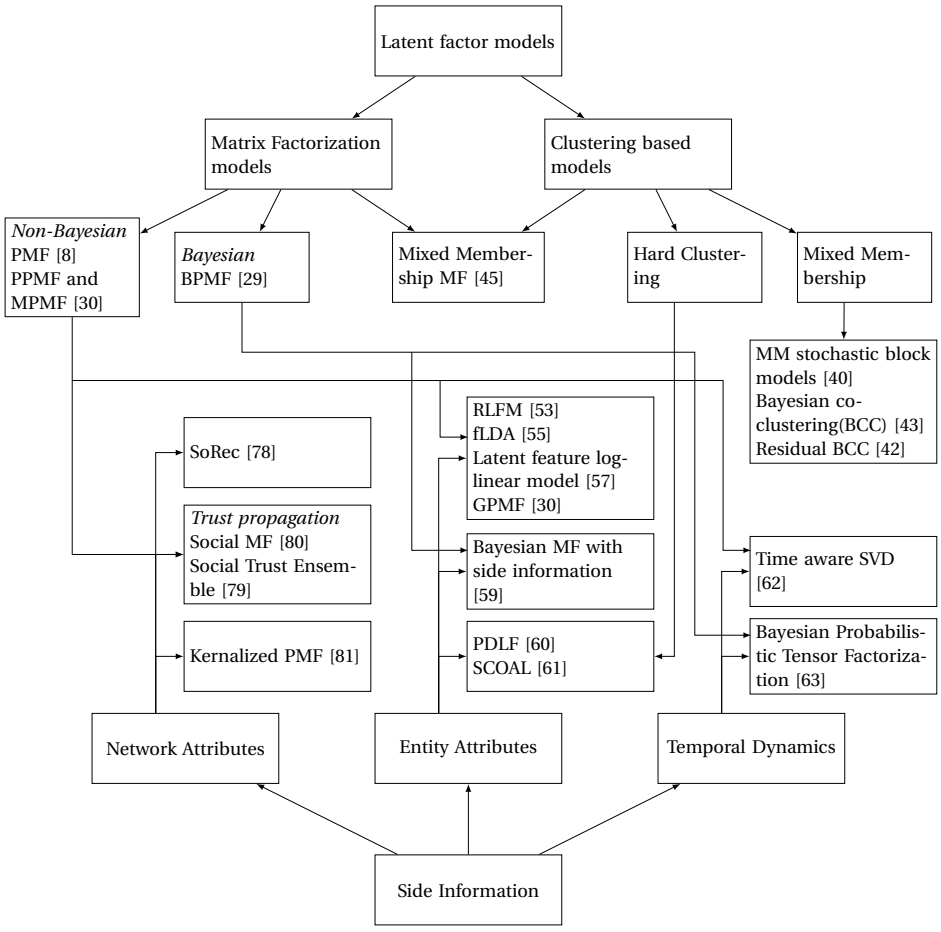


Figure 3.1: A summary relating the latent factor models discussed in the report

features corresponding to user i, item j and dyad (i, j) respectively.

3.1 Using Entity Attributes

Traditionally, several attempts were made to build hybrid recommendation systems that use user and item features as well as past interactions to combine the effects of content based and collaborative filtering. A simple approach to build such systems is to combine appropriately weighed predictions from a content based model and a CF model [50]. The weights for combining the models are learned for each user based on the strength of prediction from each model. Thus, for users with large number of ratings, the predictions from the CF model takes predominance as opposed to new users who are given recommendations based on content based model. Such models are however extremely expensive in terms of memory and computation as two models need to be built and maintained in place of one.

A popular implementation of a unified model having both the content based and CF flavors is the "Fab" [3]. *Fab* builds user profiles from web-page text features of the items "liked" by users (as in content based systems). These profiles are compared to find nearest neighbors in the system. The unknown propensity of a user for an item is calculated by combining the similarity of the item with user profile as well as the ratings given to the item by users with similar profiles. This model uses just the item profile and defaults to neighborhood based models with limited features. a straightforward extension to matrix factorization based CF methods is not available. Moreover, it is more heavily driven by content than by past interactions (as the user similarities are also content driven) and the effectiveness of the model is highly dependent on the effectiveness of the item features selected to build the user profiles.

A few publications are found in literature, that are inspired by the idea of "filterbots" for using the entity attributes in CF. The broad idea is to use content based rating agents called *filterbots* to be included as additional users in the CF framework. Sarwar et. al [17], use *filterbots* that evaluate new items upon their arrival and estimate ratings for those items based on a pre-built models (the authors use three filterbots which rates articles based on proportion of spelling error, length of article and percentage of quoted text). Users agreeing to a particular filterbot will be highly correlated with it as opposed to a user whose ratings differ drastically from that of the filterbot. The authors in [51] extend the idea proposed by Sarwar et. al. to create more sophisticated personalized filterbots. In the later work, multiple *filterbots* are built, whose combinations are personalized to each individual user. In these bots, different sets of item features (corresponding to different filterbots) are weighted by each users profile and a personalized agent for each user is inserted into a CF framework. For each user, a filterbot is generated by either choosing the best feature set or by using a weighted average of the different agents created using different sets of features. In the more recent work by [52], a reduced number of filterbots are used to get comparable performance in an item based neighborhood model. Again, a lot of of human engineering is needed to design the best set of filterbots. A very generic filterbot will show similarity with all users and highly specific ones agree with very few users.

Most of these hybrid systems had provisions to include just the item metadata and lacked the flexibility to include user and dyad features. Moreover, these models are developed over neighborhood based CF framework and cannot be trivially extended to latent factor models. As described in Chapter 2, probabilistic models based on latent factors were known to be the best performing stand-alone CF systems. Some of the successful implementations

of latent factor based CF models using entity attributes are discussed in the following subsections.

3.1.1 Regression Based Latent Factor Model

Agarwal et. al. [53], proposed a latent factor model based on PMF called "*Regression Based Latent Factor Model (RLFM)*". This incorporates the entity attributes and past interaction data into a single model for predicting the unknown affinities, thus, provides an unified framework for handling cold start and warm start scenarios. The model is based on standard probabilistic matrix factorization [8] described in Chapter 2, where the affinities are modeled as interaction of the user and the item latent factors. In PMF the user and item factors are provided a *zero* mean Gaussian prior for regularization. RLFM is similar in spirit, but rather than a zero mean Gaussian prior, the prior is a Gaussian centered around a feature based regression value. This anchors the user and item factors around a feature based model and the deviations from the feature based model is smoothly controlled by the previously observed affinity values of a given entity. It also induces marginal correlations among response values that share a common user or item which improves the predictive performance. Thus, for "light" users, with no or fewer ratings, the latent factors are driven towards the feature based estimates and for "heavy" users the estimates are appropriately refined based on the observed interactions.

Apart from user and item latent factors, user and item bias terms, $\alpha \in \mathbb{R}^{\mathbf{M}}$ and $\beta \in \mathbb{R}^{\mathbf{N}}$ respectively, are included to model the affinity values. The graphical model associated with RLFM is given in Figure 3.2.

The two stage hierarchical model to generate the affinity response for a dyad (i, j),

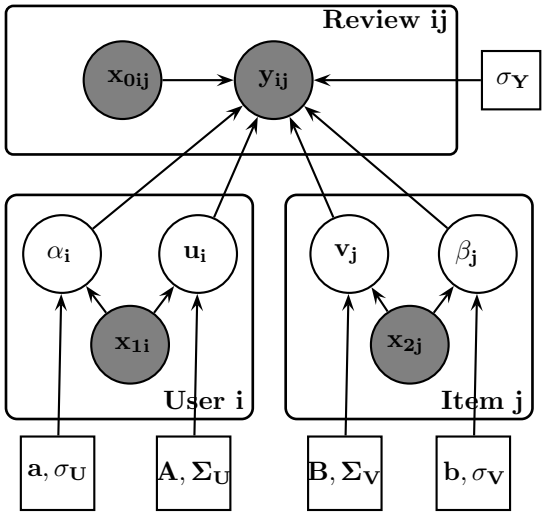


Figure 3.2: Regression Based Latent Factor Model

given the latent factors and bias is given below:

$$P(y_{ij}|X_0, X_1, X_2, \boldsymbol{u_i}, \boldsymbol{v_j}, \alpha_i, \beta_j, \boldsymbol{c}, \sigma_Y) = \mathcal{N}(y_{ij}|\boldsymbol{c}^T \boldsymbol{x_{0ij}} + \alpha_i + \beta_j + \boldsymbol{u_i}^T \boldsymbol{v_j}, \sigma_Y^2) \tag{3.1}$$

The priors on U, V, $\boldsymbol{\alpha}$, $\boldsymbol{\beta}$ are given by:

$$P(\boldsymbol{u_i}|\boldsymbol{x_{1i}}, A, \Sigma_U) = \mathcal{N}(\boldsymbol{u_i}|A\boldsymbol{x_{1i}}, \Sigma_U) \tag{3.2}$$

$$P(\boldsymbol{v_i}|\boldsymbol{x_{2j}}, B, \Sigma_V) = \mathcal{N}(\boldsymbol{v_i}|B\boldsymbol{x_{2j}}, \Sigma_V) \tag{3.3}$$

$$P(\alpha_i|\boldsymbol{x_{1i}}, \boldsymbol{a}, \sigma_U) = \mathcal{N}(\alpha_i|\boldsymbol{a}^T\boldsymbol{x_{1i}}, \sigma_U^2) \tag{3.4}$$

$$P(\beta_i|\boldsymbol{x_{2j}}, \boldsymbol{b}, \sigma_U) = \mathcal{N}(\beta_i|\boldsymbol{b}^T\boldsymbol{x_{2j}}, \sigma_V^2) \tag{3.5}$$

where $\boldsymbol{a} \in \mathbb{R}^{D_1}$ and $A \in \mathbb{R}^{D \times D_1}$, $\boldsymbol{b} \in \mathbb{R}^{D_2}$ and $B \in \mathbb{R}^{D \times D_2}$, and $\boldsymbol{c} \in \mathbb{R}^{D_0}$ are regression coefficients associated with the user, item and cell features respectively, and $\Sigma_U \in \mathbb{R}^{D \times D}$, $\Sigma_V \in \mathbb{R}^{D \times D}$ are the covariance matrices associated with the user and item latent factors respectively. Finally, σ_U^2 and σ_V^2 are the variances associated with user and item biases respectively.

The likelihood of observed and latent variables is given by:

$$P(Y,U,V,\boldsymbol{\alpha},\boldsymbol{\beta}|X_0,X_1,X_2,\Theta) = \prod_{(ij)\in\kappa} \mathcal{N}(y_{ij}|\boldsymbol{c}^T x_{0ij} + \alpha_i + \beta_j + u_i^T v_j, \sigma_Y^2) \prod_{i=1}^{M} \{\mathcal{N}(\boldsymbol{u_i}|A\boldsymbol{x_{1i}}, \Sigma_U)\mathcal{N}(\alpha_i|\boldsymbol{a}^T\boldsymbol{x_{1i}}, \sigma_U^2) \prod_{i=1}^{M} \{\mathcal{N}(\boldsymbol{v_j}|B\boldsymbol{x_{2j}}, \Sigma_V)\mathcal{N}(\beta_j|\boldsymbol{b}^T\boldsymbol{x_{2j}}, \sigma_V^2) \tag{3.6}$$

where $\Theta = \{\Theta_1, \Theta_2\}$ are the model parameters and $\Theta_1 = \{\boldsymbol{c}, \boldsymbol{\sigma_Y}\}$ are the parameters of the first stage and $\Theta_2 = \{\boldsymbol{a}, \boldsymbol{b}, A, B, \sigma_U, \sigma_V, \Sigma_U, \Sigma_V\}$ are the parameters of the second stage

The model is learned through an Expectation Maximization algorithm that finds a local optimum by alternatively computing the expected marginal observed log likelihood (E-Step) and updating the hyperparameters to maximize the expected marginals (M-Step) [54]. As with BPMF, estimating the exact posterior of observed variables given the latent variable is intractable and is approximated by MCMC simulations through Gibbs sampling. This inference algorithm is easily parallelizable (as in case of BPMF), leading to scalable implementation. The prediction is estimated through the ML estimate, from Equation 3.1.

The experiments on standard datasets show significant improvement in accuracy for new users in the presence of entity features and marginal improvement in prediction accuracy for old users. However, owing to increased number of parameters to fit, the model is fairly complex and is highly prone to overfitting in the absence of sufficient data. Further, there is no implicit feature selection and hence, tedious feature engineering over the available entity features is required to avoid overfitting.

3.1.2 fLDA

fLDA is a matrix factorization model proposed by Agarwal et. al. [55], which is similar in flavor to the previously described RLFM model [53]. This model is intended for systems that have "bag-of-words" representation for the item metadata. In such a case, the item latent factors, $\boldsymbol{v_j}$ are derived from the proportions of latent "topics" present in the item meta data and the user latent factors, $\boldsymbol{u_i}$ serve as affinities of the user for the latent topics. This idea is inspired from the *latent Dirichlet allocation (LDA)* [41] models for bag-of-words data and its supervised version, sLDA [56]. The latent topic distribution of the item meta data is obtained as an average of topic assignments for the individual words. As in LDA, for each item j, a distribution over topics, $\boldsymbol{\theta_j}$ is drawn from a Dirichlet distribution parameterized by $\boldsymbol{\lambda}$ and the words are drawn from the topic-assignments sampled from $\boldsymbol{\theta_j}$. The total number of words in the vocabulary is denoted as W and the number of words in the metadata of item j is denoted by W_j. The word-topic distributions are given by the matrix $\Phi \in \mathbb{R}^{W \times K}$, where K is the number of topics. The word distribution for topic, k, given by $\boldsymbol{\phi_k} \in \mathbb{R}^W$, is drawn from a Dirichlet distribution parameterized by $\boldsymbol{\eta}$. The priors on user latent factors, $\boldsymbol{u_i}$ and the user and item biases, α_i, β_j are obtained from a regression model over the user and item metadata as in RLFM (Equation 3.2). The generative Process is described in Model 4 The graphical model is given in Figure 3.3. The model parameters are learned through ML estimate of the observed ratings and words in item metadata. In fLDA, unlike in traditional LDA, the user factors and item topic distributions are simultaneously learned to better predict the response variable. Thus, the topics learned are predictive of the affinity response. The inference is performed using MCMC based EM algorithm.

In the presence of rich text data, the model performs significantly better than exist-

Model 4 Generative Process for fLDA

for Each topic, $k = 1, 2, \ldots, K$ **do**
  Sample $\phi_k \sim Dir(\eta)$
end for
for $i = 1, 2, \ldots, M$ **do**
  Sample $\alpha_i \sim \mathcal{N}(a^T \boldsymbol{x_{1i}}, \sigma_U^2)$
  Sample $\boldsymbol{u_i} \sim \mathcal{N}(A\boldsymbol{x_{1i}}, \Sigma_U)$
end for
for $j = 1, 2, \ldots, N$ **do**
  Sample $\beta_j \sim \mathcal{N}(b^T \boldsymbol{x_{2j}}, \sigma_V^2)$
  Sample $\boldsymbol{\theta_j} \sim Dir(\boldsymbol{\lambda})$
  for Each word in item j, $n = 1, 2, \ldots, W_j$ **do**
    Sample $z_{jn} \sim Mult(\boldsymbol{\theta_j})$
    Sample $w_{jn} \sim Mult(\boldsymbol{\phi_{z_{jn}}})$
  end for
  $v_j = \sum_n z_{jn}/W_j$
end for
for $(i, j) \in (M \times N)$ **do**
  Sample $y_{ij} \sim \mathcal{N}(\alpha_i + \beta_j + \boldsymbol{u_i}^T \boldsymbol{v_j}, \sigma_Y^2)$
end for

ing matrix factorization methods, RLFM and filterbots based methods. As LDA inherently is a feature selection technique, the topic proportions are fairly good indicators of item profiles, this inherent feature selection results in better performance when compared to very similar RLFM. Also as an auxiliary result, fLDA results in interpretable topics that explain the user affinities.

3.1.3 Latent Feature log-linear (LFL) Model

The models discussed so far are generative models (that model the joint probability distribution of observed and unobserved variables). Another popular choice is to learn discriminative models (that model just the conditional distribution of the unobserved variables conditioned on the observed ones) for affinity prediction. A discriminative model for response prediction in dyadic data was proposed in [57]. The authors propose a model in

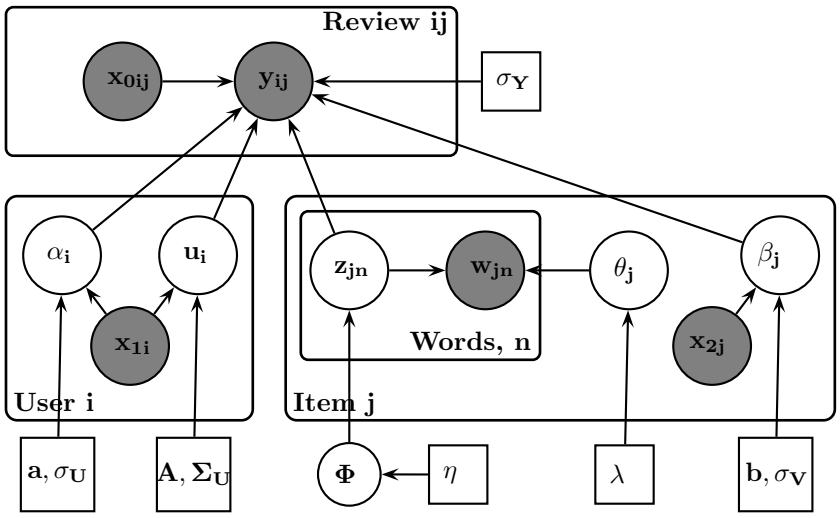


Figure 3.3: fLDA

which multiple kinds of response– like ordinal, numeric and nominal– are modeled in a single framework with provisions to include side information. The model approximates the log of conditional distribution of the affinity of a dyad, y_{ij} by a low rank matrix. For a dyad (i, j), the concatenation of the user, item and dyad metadata is denoted by:

$$\boldsymbol{x_{ij}} = [\boldsymbol{x_{0ij}}, \boldsymbol{x_{1i}}, \boldsymbol{x_{2j}}]$$

The user and item factor matrices are three way tensors, $U \in \mathbb{R}^{M \times D \times |\mathcal{Y}|}$ and $V \in \mathbb{R}^{N \times D \times |\mathcal{Y}|}$ respectively, where $\mathcal{Y}$ is the set of possible values, the response variables, $\{y_{ij}\}$ can take. For each value of $y \in \mathcal{Y}$, the conditional distribution of the response for a dyad (i, j) is given by:

$$\log P(y_{ij} = y | x_{i,j}, U, V, W) = \langle \boldsymbol{u_i^{(y)}}, \boldsymbol{v_j^{(y)}} \rangle \; + \boldsymbol{w_{ij}}^T \boldsymbol{x_{ij}} \; + constant \tag{3.7}$$

where $\boldsymbol{u_i^{(y)}} = U[i, :, y]$ and $\boldsymbol{v_j^{(y)}} = V[j, :, y]$ quantifies the tendency of ith user (and jth item) to give (or receive) an affinity value of y along the D latent dimensions, and $\mathbf{w_{ij}}$ is the weight vector for the meta information present. For numeric response types, the prediction

for a new dyad pair is estimated as the mean of the above conditional distribution (median for non-numeric response), as it is also the maximum likelihood estimate.

$$\hat{y}_{ij} = \mathbb{E}_y[P(y_{ij} = y|x_{ij}, U, V, W)]$$

The model parameters $U \in \mathbb{R}^{M \times D \times |\mathcal{Y}|}$, $V \in \mathbb{R}^{N \times D \times |\mathcal{Y}|}$ and $W \in \mathbb{R}^{M \times N}$ are learned by minimizing the regularized squared loss given below:

$$L(U, V, W) = \sum_{(ij) \in \kappa} (y_{ij} - \hat{y}_{ij})^2 + \lambda(\|U\|_F^2 + \|V\|_F^2 + \|W\|_F^2) \tag{3.8}$$

As the objective is differentiable, the minimization can be performed by scalable Stochastic Gradient Descent(SGD) algorithm.

The main contribution of the work is a simple discriminative framework for modeling various response types with provision for inclusion of side information in a systematic manner. For the task of link prediction, the experimental results show a significant improvement over the existing methods.

However, the model suffers from the drawback of tendency to overfit owing to large number of parameters to be learned ($\mathcal{Y}$ times the number of parameters for vanilla PMF). Various extensions of the above model were also proposed in the paper to overcome the problem, including using different regularizers for users, items and metadata weights. The authors also propose a reduced parameter version for ordinal numeric response types which allows for parameter sharing across different values of y.

3.1.4 Generalized Probabilistic Matrix Factorizations(GPMF) with Side Information

In Section 2.1.3, two Generalized PMF models [30], PPMF and MPMF, were discussed. The extensions of these models to include side-information from the entity at-

tributes, is described in this subsection. The idea is to use topic models over the side-information and PMF over the ratings matrix. The coupling between two models is established through the shared latent variables. The authors propose two models, "Correlated Topic Model PPMF (CTM-PPMF)" and "Latent Dirichlet Allocation MPMF (LDA-MPMF)", to include item metadata in the form of bag-of-words.

In CTM-PPMF, the item latent factors, $\mathbf{v_j}$ also serve as the inferred distribution topics in item's metadata. These topics in turn generate the words in the item metadata through the generative process of a correlated topic model. A correlated topic model [58], is an extension of LDA [41], where the topic proportions learned exhibit correlation via a logistic normal distribution. The number of topics is denoted by K and w_{jn} denotes the nth word in the metadata of item j. Mathematically, the model follows the PPMF for generation of Y, U and V (refer to Section 2.1.3), but apart from explaining the observed ratings, the logistic function of item factors v_j also generates the bag-of-words features of the item j as follows:

- For each word, w_{jn} in the bag-of-words of item j side information
    1. Sample a topic $z_{jn} \sim Mult(logistic(v_j))$
    2. Sample a word $w_{jn} \sim Mult(\phi_{z_{jn}})$

where, ϕ_k is the distribution of words over the topic k.

In LDA-MPMF, the kth Gaussian of the mixture-of-Gaussian (MOG) prior of the MPMF models also serves as the distribution of item latent factor for an item, whose metadata is generated from the topic k with probability 1. Thus, latent variables, ρ_j, which are also the Dirichlet discrete prior mixing proportions for the mixture of Gaussian

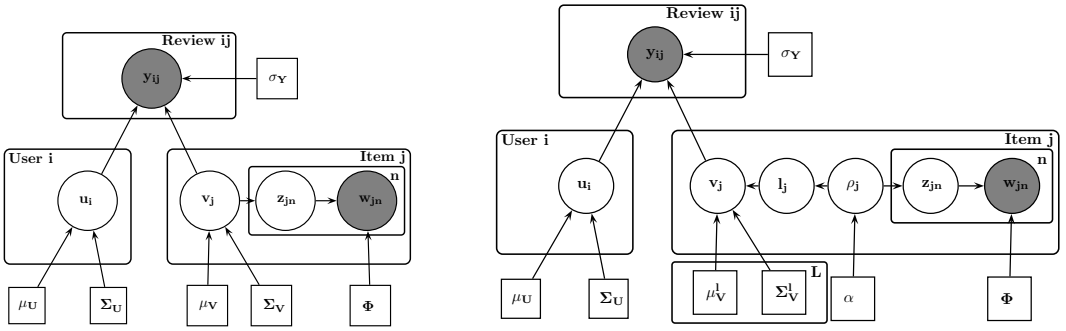
is also shared as the topic proportions for the bag-of-words representation of the item. The generative process for Y, U, V for LDA-MPMF is the same as that of MPMF (refer to Algorithm 2 for the generative process and the notations), and the words of the item metadata for item j are generated as follows:

- For nth word w_{jn} in the bag-of-words of item j
    1. Sample a topic $z_{jn} \sim Mult(\boldsymbol{\rho_j})$
    2. Sample a word $w_{jn} \sim Mult(\phi_{z_{jn}})$

where, ϕ_k is again the distribution of words over the topic k. The generative process for CTM-PPMF and LDA-MPMF are given by the graphical models in Figures 3.4a and 3.4b respectively.

The idea behind these models is that the items with similar side information tend to have similar topic distributions which in turn leads to similar item latent factors (in case of CTM-PPMF, this is achieved through similar $logisctic(v_j)$ values and in LDA-MPMF, this is derived from similar items choosing the same Gaussian with high probability for generating v_j).

The experimental results showed better performance of CTM-PPMF and LDA-MPMF compared to PPMF and MPMF. Overall, the PPMF models (PPMF and CTM-PPMF) performed better than MPMF models (MPMF and LDA-MPMF). These models showed improvement over the vanilla PMF and BPMF and other co-clustering based algorithms. But comparison with other models incorporating side-information like RLFM [53] and fLDA [55] are not provided.



(a) CTM-PMF (b) LDA-MPMF

Figure 3.4: Generalized PMFs with Side Information

3.1.5 Bayesian Matrix Factorization with Side Information (BMFSI)

A complete Bayesian treatment of matrix factorization with side information was provided by Porteous et. al. [59]. The idea is to extend the BPMF [29] to include side information by performing regression against the user, item and dyad metadata. Unlike RLFM, the latent factors and regression coefficients for regression are treated jointly as described below. Moreover, the Bayesian treatment allows for automatic complexity control to avoid overfitting at the cost of increased computation.

In *Bayesian matrix factorization with side information (BPMFSI)*, the user and item latent factors are augmented as described below.

- For each dyad (i, j)

  1. $\boldsymbol{u_{ij}} = [\boldsymbol{u_i}\ \boldsymbol{u_{ai}}\ \boldsymbol{x^u_{ij}}]$ is a concatenation of the user latent factor, $\boldsymbol{u_i}$, regression coefficients for the item metadata, $\boldsymbol{u_{ai}}$ and the observed user features (for the dyad (i,j)), $\boldsymbol{x^u_{ij}}$, where $\boldsymbol{x^u_{ij}}$ is essentially a subset of user and dyad side information, $\boldsymbol{x^u_{ij}} \subset \{\boldsymbol{x_{1i}}, \boldsymbol{x_{0ij}}\}$.

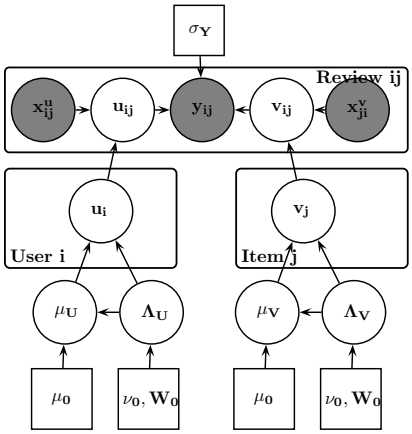


Figure 3.5: Bayesian Matrix Factorization with side information

2. Similarly, $\boldsymbol{v_{ij}} = [\boldsymbol{v_j}\ \boldsymbol{x^v_{ji}}\ \boldsymbol{v_{bj}}]$

The mean of the predictive distribution is given by:

$$\begin{aligned} \mu_{ij} &= \boldsymbol{u_{ij}}^T \boldsymbol{v_{ij}} \\ &= \boldsymbol{u_i}^T \boldsymbol{v_j} + \boldsymbol{u_{ai}}^T \boldsymbol{x^v_{ji}} + \boldsymbol{x^u_{ij}}^T \boldsymbol{v_{bj}} \end{aligned} \tag{3.9}$$

The model for response value is given by:

$$y_{ij} \sim \mathcal{N}(\mu_{ij}, \sigma_Y^2) \tag{3.10}$$

The latent factors, $\boldsymbol{u_i}, \boldsymbol{u_{ai}}, \boldsymbol{v_j}, \boldsymbol{v_{bj}}$ are given Gaussian-Wishart conjugate prior as in BPMF (refer Equations 2.8 and 2.9). The generative process is described in the graphical model in Figure 3.5.

The authors also extend this model to introduce a richer Dirichlet process mixture-of-Gaussian prior on the user and item latent factors. This in turn provides higher flexibility by giving different regularization for each of the D latent classes. The BMFSI model assumes that every user and movie draws latent factors from a single multivariate Gaussian. To

explain the heterogeneity in the data, a more generalized prior of mixture of Gaussian was used for both user and item latent factors. Also, to make the model non-parametric, a Dirichlet process mixture was used which allows for countably infinite number of mixture components. In the generative process for this new model, instead of drawing a factor vector from a single common distribution, each user or item first draws an index and then draws the latent factor from the Gaussian with that index.

These models are learned through MCMC methods using Gibbs Sampling. As for the performance, the experiments show that the model significantly outperforms the BPMF and other baseline approaches.

3.1.6 Predictive Discrete Latent Factor Model (PDLF)

A clustering based models that uses side information was proposed by Agarwal et. al. [60]. The *Predictive Discrete Latent Factor (PDLF)* model proposed in the paper simultaneously incorporates the effect of the covariates via a global model and captures local structure through a co-cluster specific constant. In this model, the response matrix is partitioned into a grid of co-clusters representing local regions. The dyadic response is modeled as sum of functions of available covariate information and unmeasured latent factors obtained through co-clustering.

The set of covariates associated with a dyad (i, j), are given by, $\boldsymbol{x_{ij}} = [\boldsymbol{x_{0ij}}, \boldsymbol{x_{1i}}, \boldsymbol{x_{2j}}]$. K and L are the number of row and column clusters respectively. Each of the KL co-clusters are assigned a global prior probabilities given by $\boldsymbol{\pi} \in \mathbb{R}^{KL}$, such that π_{kl} is the probability of a random point in response matrix Y being associated with the co-cluster, (k, l). The

response for a dyad (i, j) is modeled as:

$$P(y_{ij}|\boldsymbol{x_{ij}}) = \sum_{kl} \pi_{kl} p_{\psi}(y_{ij}|\boldsymbol{\beta}^T \boldsymbol{x_{ij}} + \delta_{kl}) \tag{3.11}$$

where $P_{\psi}(y|\theta)$ is a exponential family distribution with parameter θ, $\boldsymbol{\beta}$ represents the regression coefficients for the covariates and δ_{kl} is the interaction effect captured by the cluster membership. Also, unlike traditional "divide and conquer" approaches, in PDLF the covariate effects and co-cluster assignments are carried out simultaneously, which leads to more optimal solutions than clustering apriori and then building predictive models. The authors also formulate scalable, generalized EM based algorithms to estimate the parameters of hard and soft versions of the proposed model. The model can be adapted for prediction of various kinds of responses (binary, nominal, continuous etc).

3.1.7 Simultaneous Co-clustering And Learning (SCOAL)

Simultaneous Co-clustering and Learning (SCOAL), described in [61], is a framework for dealing with dyadic data with covariates (DyadCs). The core idea is to co-cluster the dyads while simultaneously building predictive models within each co-cluster. The objective of the model is to obtain a partitioning of rows and columns such that the response values in each co-cluster can be well characterized by a single predictive model. The framework is characterized by a global objective function and the task is to estimate co-cluster assignments and the co-cluster specific models that minimizes the global objective.

In the context of recommendation systems, the global objective is suitably regularized sum squared error over observed entries. The predictive models withing each co-cluster is restricted to be a generalized linear models over the covariates. The task reduces to finding mappings $\rho : \{1, 2, \ldots, M\} \rightarrow \{1, 2, \ldots, K\}$ and $\gamma : \{1, 2, \ldots, N\} \rightarrow \{1, 2, \ldots, L\}$,

and model parameters for each co-cluster, $\boldsymbol{\beta_{kl}}$, that minimizes the global objective function. Mathematically, the problem is formulated as below:

$$\rho^*, \gamma^*, \boldsymbol{\beta^*} = \underset{\rho,\gamma,\boldsymbol{\beta}}{argmin} \sum_{k=1}^{K}\sum_{l=1}^{L}\left(\sum_{i:\rho(i)=k}\sum_{j:\gamma(j)=l} I_{ij}(y_{ij} - \boldsymbol{\beta_{kl}}^T \boldsymbol{x_{ij}})^2 + \lambda\|\boldsymbol{\beta_{kl}}\|_2^2\right) \tag{3.12}$$

A simple iterative algorithm that alternately updates the co-cluster models and the row and column cluster assignments can be applied to obtain a local minimum of the objective function. The algorithm is described in Algorithm 5

Algorithm 5 Iterative updates for SCOAL

Randomly initialize ρ, γ
repeat
  for $k = 1, 2, \ldots, K$ **do**
    for $l = 1, 2, \ldots, L$ **do**
      Update $\beta_{kl} = \underset{\beta}{argmin} \sum_{i:\rho(i)=k}\sum_{j:\gamma(j)=l} I_{ij}(y_{ij} - \boldsymbol{\beta}^T \boldsymbol{x_{ij}})^2 + \lambda\|\boldsymbol{\beta}\|_2^2$
    end for
  end for
  $\forall i$, Update $\rho(i) = \underset{k}{argmin} \sum_{l=1}^{L}\sum_{j:\gamma(j)=l} I_{ij}(y_{ij} - \boldsymbol{\beta_{kl}}^T \boldsymbol{x_{ij}})^2 + \lambda\|\boldsymbol{\beta_{kl}}\|_2^2$
  $\forall j$, Update $\gamma(j) = \underset{l}{argmin} \sum_{k=1}^{K}\sum_{i:\rho(i)=k} I_{ij}(y_{ij} - \boldsymbol{\beta_{kl}}^T \boldsymbol{x_{ij}})^2 + \lambda\|\boldsymbol{\beta_{kl}}\|_2^2$
until convergence

3.2 Using Temporal Dynamics

The matrix factorization methods considered so far assume the system to be static and ignores the temporal dynamics of the ratings. However, the preferences of entities are inherently dynamic in nature. The popularity of items frequently change over time and with the advent of new items. Further, the user preference factors and their bias change over time too. Accurate modeling of these dynamics can greatly influence the accuracy of prediction of propensities at a future time point.

3.2.1 Time Aware SVD (timeSVD)

Koren et. al. [27, 62] introduced various temporal aspects into a matrix factorization framework through a timeSVD model which was applied on the Netflix dataset. They consider the basic matrix factorization model that accounts for biases given by the Equation 2.2. The equation is repeated for reference:

$$y_{ij} = \mu + \alpha_i + \beta_j + \boldsymbol{u_i}^T \boldsymbol{v_j} \tag{3.13}$$

where, μ is the global average of the ratings, α_i and β_j are the user and item bias respectively. The authors model the temporal effects of each factor separately. The effect of varying popularity of the items is incorporated in the item bias β_j. The item bias is now modeled as a sum of static bias and a time varying component which varies coarsely over the time bins of size Δ, such that $\Delta(t)$ represent the bin index containing the time t, $\beta_j(t) = \beta_j + \beta_j(\Delta(t))$. On the contrast, the user bias towards rating items and user factors influencing the ratings are varied smoothly with time as $\alpha_i(t)$ and $\boldsymbol{u_i}(t)$. These continuous effects are captured by smoothly varying parameterized functions such as $f(t; \gamma, t_0) = f_0 + sgn(t - t_0)\|t - t_0\|^\gamma$. Further, the periodic and daily effects where appropriately accounted for in these factors. The item latent factors remain static following the intuition that the item characteristics are static with time. In the time changing factor model timeSVD, the ratings are now modeled as:

$$y_{ij}(t) = \mu + \alpha_i(t) + \beta_j + \beta_j(\Delta(t)) + \boldsymbol{u_i}(t)^T \boldsymbol{v_j} \tag{3.14}$$

The model parameters are learned by minimizing the regularized squared error loss on the above estimate. The results show improvement over their static counterparts. Though the method incorporates various dynamic effects in a intuitive framework, the model

involves a large number of parameters and accurate learning of these parameters depends heavily of the availability of sufficient training data. However, the real life datasets are highly sporadic and sparse that are prone to overfitting in this model.

3.2.2 Bayesian Probabilistic Tensor Factorization (BPTF)

The *Bayesian Probabilistic Tensor Factorization (BPTF)* [63], incorporates the temporal effects through a smaller number of parameters as compared to timeSVD [62] previously described. The time is discretized into time slices indexed by k. The ratings are now modeled as a three way factor involving the users factors $\mathbf{u_i} \in \mathbb{R}^D$, item factors, $\mathbf{v_j} \in \mathbb{R}^D$ and the time varying factors, $\mathbf{t_k} \in \mathbb{R}^D$. The user and item latent factors are learned as functions of time. The model for rating y_{ij} in the time slice k, in the tensor factorization is given below:

$$P(y_{ij}(k)|\mathbf{u_i}, \mathbf{v_j}, \mathbf{t_k}) = \mathcal{N}(y_{ij}|\langle \mathbf{u_i}, \mathbf{v_j}, \mathbf{t_k} \rangle, \sigma_Y^2) \tag{3.15}$$

where $\langle \mathbf{u_i}, \mathbf{v_j}, \mathbf{t_k} \rangle = \sum_{d=1}^{D} u_{id} v_{jd} t_{kd}$. The priors on the user and item latent factors are similar to the BPMF model [29].

$$\begin{aligned} P(\mathbf{u_i}|\boldsymbol{\mu_U}, \Lambda_U) &= \mathcal{N}(\boldsymbol{u_i}|\boldsymbol{\mu_U}, \Lambda_U^{-1}) \\ P(\mathbf{v_j}|\boldsymbol{\mu_V}, \Lambda_V) &= \mathcal{N}(\boldsymbol{v_i}|\boldsymbol{\mu_V}, \Lambda_V^{-1}) \end{aligned} \tag{3.16}$$

For the time factors, a smooth variation is imposed using a Markovian assumption, wherein, the time factor at k depends only on the time factor in the previous slice $k-1$. The Bayesian priors are given by:

$$\begin{aligned} P(\mathbf{t_k}|\boldsymbol{t_{k-1}}, \Lambda_T) &= \mathcal{N}(\boldsymbol{t_k}|\boldsymbol{t_{k-1}}, \Lambda_T^{-1}) \; \forall k > 1 \\ P(\mathbf{t_1}|\boldsymbol{\mu_T}, \Lambda_T) &= \mathcal{N}(\boldsymbol{t_k}|\boldsymbol{\mu_T}, \Lambda_T^{-1}) \end{aligned} \tag{3.17}$$

Further, the overfitting issue is avoided by considering a full Bayesian treatment of the tensor factors. For a complete Bayesian treatment, the hyperparameters $\alpha = \frac{1}{\sigma_Y^2}$,

$\Theta_U = \{\boldsymbol{\mu_U}, \Lambda_U)\}$, $\Theta_V = \{\boldsymbol{\mu_V}, \Lambda_V)\}$ and $\Theta_T = \{\boldsymbol{\mu_T}, \Lambda_T)\}$ are assigned Gaussian-Wishart conjugate priors similar to those in Equation 2.9. An MCMC based sampling algorithm is developed to learn the parameters of the model.

In comparison to timeSVD, while timeSVD captures the local effects of time on preferences, the BPTF models global effect of time that are shared across the users and this leads to drastic reduction in number of parameters involved. The model showed improvement over the static models, however for Netflix dataset, the timeSVD model provided better results as it was able to capture various local temporal effects of the dataset. This model is motivated from a very similar idea proposed by Hoff [64] for factoring multi-way data.

3.3 Using Network Structure

Social networks serve as one of the most effective channels for dissemination of information. These networks can provide useful information about user profiles which are not otherwise explicitly available. Moreover, traditional recommender systems assume independence among the users. This assumption is violated in the presence of known social interactions or connections. The idea is to use the additional structure and information in these networks to enhance performance of recommendation systems. In reality, users' explicit social network has a strong influence in modeling the user preference and hence ignoring such information is sub-optimal. The initial work in this direction mainly applied network information to memory based CF methods. These are briefly described first following which a detailed description of models for using social network information in latent factor based CF systems is provided.

The following additional notation is used in this section. The social network graph is denoted by $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ denotes the set of nodes in the network with individual nodes indexed as v_i, and $\mathcal{E}$ denotes the set of edges in the graph with e_{ik} denoting the directed edge from v_i to v_k. Let $C = \{c_{ik}\}$ denote the adjacency matrix of $\mathcal{G}$, where the entries $0 \leq c_{ik} \leq 1$ represent the strength of the edge from node v_i to v_k if e_{ik} exists and 0 otherwise. The weight of the edge quantifies the trust of the head node on the tail node. Finally, $N_i = \{v_k \in \mathcal{V} : e_{ik} \in \mathcal{E}\}$ are the set of nodes trusted by node i

In one of the earliest work, Referral Web [65], the underlying network is formed by using the co-occurrences of the names of its users in publicly accessible world wide web documents, and links (or referrals) are recommended from the underlying network. Papagelis et. al. [66], proposed one of the simplest models for incorporating social network information in neighborhood based CF techniques. This is known to be specially effective when the sparsity in the ratings matrix is high. The model aims at augmenting the available data for user based neighborhood models. An initial network of the users in the system is constructed based on similarities of the users (obtained directly from the rating matrix). This initial graph essentially has an edge between a pair of users whenever they have rated common items and the strength of the links, also known as the "trust" values, are computed through one of the many existing similarity measures. Moreover, the strength in each link is measured by a "confidence" metric that is proportional to the number of common items rated by a pair of users and is normalized to a value between 0 and 1. The network is then augmented by adding edges to 2-hop and 3-hop neighbors. The trust in these new links are compositions of the trust values of the links in the path leading the source node to the target node and the confidence is simply the product of the confidence values of

the path. This kind of inference of new links from the existing ones is known as *trust inference* or *trust propagation*. Though the work does not use the social network as a "side-information", it is trivial to include external source of network information as the initial or the augmented network. The model showed significant improvement in performance of user based neighborhood methods when the augmented network of user similarities (2-hop and 3-hop neighbors) is used. However, the improvement saturates at 3-hop for all sparsity levels. A work very similar to this using external side information is SNACK [67], [68]

Another model for using social networks in neighborhood based CF systems was outlined in [69] and later formulated in [70]. In this model, the underlying bipartite graph between users and movies is used to compute the similarities between items and/or users. User nodes with large number of short path lengths imply that the pair of users have watched many similar movies and hence should be more similar. The traditional distance metrics like shortest path do not have the property of decreasing the distance when new links are added. The authors use a Markov chain random walk models [71] to the graph to compute new distance metrics that have the above mentioned desirable properties. Essentially, a "random walker" starts at a node i at time $t = 0$ and at each time step, the walker proceeds to a neighboring node with transition probability proportional to the weight of the link (independent of the previous nodes visited). The *average first-passage time* from a node i to a node j, $m(j|i)$ is defined as the average number of steps needed by a random walker for reaching state j for the first time having started from node i. The *average commute time*, given by $n(i, j) = m(i|j) + m(j|i)$ serves as a distance measure between any pair of states. This measure is also related to *Euclidean Commute Time Distance* [72][1] and the

[1]distance between the nodes in the Euclidean space spanned by the rows of the adjacency matrix

Moore Penrose pseudo inverse of the Laplacian matrix[2] of the graph. These similarity metrics, as opposed to traditional methods, have the desirable property of increasing when the number of paths connecting the elements increases and when the length of paths decrease. Experimental results on the MovieLens database show that the Laplacian-based similarities perform well in comparison with the other existing similarity measures.

The other methods that used social network to improve the performance of neighborhood based methods include [73] and [74] that use trust based methods for similarity estimation. Various trust metrics and their improvements are suggested in [73, 75, 76]. An overview of methods for using social network analysis in neighborhood models is provided in [77] and the references therein.

The recent methods that use social network information in latent factor model based CD systems is discussed in the following subsections.

3.3.1 SoRec: Social Recommendation

The first work to consider incorporating external social network into a PMF framework was proposed by Ma et. al [78]. The authors propose a method for integrating user social network structure and user-item rating matrix by sharing user latent factors. Further, they provide complexity analysis of their model to indicate that the model scales linearly with the number of observations and hence is scalable to large datasets. In this model the both the adjacency matrix C and the response matrix Y are simultaneously approximated

[2]Laplacian matrix of the graph with adjacency matrix A is given by L=D-A, where D is a diagonal matrix with $d_{ii} = \sum_j a_{ij}$. The Moore Penrose pseudo inverse of L, L^+ is a matrix such that $LL^+L = L$, $L^+LL^+ = L^+$, $(LL^+)^* = LL^+$ and $(L^+L)^* = L^+L$. It is later shown that the elements of L^+ represent the inner product between nodes projected onto a subspace where ECTD is preserved and hence serves as a valid similarity measure

by low dimensional matrices UZ^T and UV^T respectively. Here $Z \in \mathbb{R}^{M \times D}$, U, V are low dimensional factors representing the network, user, and item latent factors. The generative process for the adjacency matrix C and response matrix Y given the latent variables is given by:

$$P(C|U, Z, \sigma_C^2) = \prod_{i=1}^{M} \prod_{k=1}^{M} [\mathcal{N}(c_{ik}^*|g(\boldsymbol{u_i}^T \boldsymbol{z_k}), \sigma_C^2)]^{I_{ik}^C} \tag{3.18}$$

$$P(Y|U, V, \sigma_Y^2) = \prod_{i=1}^{M} \prod_{j=1}^{N} [\mathcal{N}(y_{ij}|\boldsymbol{u_i}^T \boldsymbol{v_j}, \sigma_Y^2)]^{I_{ij}} \tag{3.19}$$

where, I_{ik}^C is an indicator variable for presence of an edge, e_{ik} in $\mathcal{G}$, I_{ij} is the indicator variable for observing the response y_{ij} and $c_{ik}^* = \sqrt{\frac{indegree(v_k)}{outdegree(v_i)+indegree(v_k)}} c_{ik}$ is the product of "trust" of user i on user k and the "confidence" in the trust. Also $g(x) = 1/(1 + e^{-x})$ is the logistic function.

The user, item and network factors are given priors as:

$$P(U|\sigma_U^2) = \prod_{i=1}^{M} \mathcal{N}(u_i|0, \sigma_U^2) \tag{3.20}$$

$$P(Z|\sigma_Z^2) = \prod_{i=1}^{M} \mathcal{N}(z_k|0, \sigma_Z^2) \tag{3.21}$$

$$P(V|\sigma_V^2) = \prod_{j=1}^{N} \mathcal{N}(v_j|0, \sigma_V^2) \tag{3.22}$$

The graphical model is shown in Figure 3.6. The model parameters are denoted by $\Theta = \{\sigma_U^2, \sigma_V^2, \sigma_Y^2, \sigma_Z^2, \sigma_C^2\}$. The latent factors are learned by maximizing the log posterior of latent variables given the observations:

$$\begin{aligned} \ln P(U, V, Z|Y_\kappa, C, \Theta) = & -\frac{1}{2\sigma_Y^2} \sum_{(ij) \in M \times N} I_{ij} (y_{ij} - \boldsymbol{u_i}^T \boldsymbol{v_j})^2 \\ & - \frac{1}{2\sigma_C^2} \sum_{(ik) \in M \times M} I_{ik}^C (c_{ik} - g(\boldsymbol{u_i}^T \boldsymbol{z_k}))^2 \\ & - \frac{1}{2\sigma_U^2} \|U\|_F^2 - \frac{1}{2\sigma_V^2} \|V\|_F^2 - \frac{1}{2\sigma_Z^2} \|Z\|_F^2 + Constant \end{aligned} \tag{3.23}$$

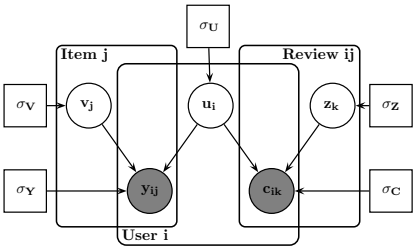
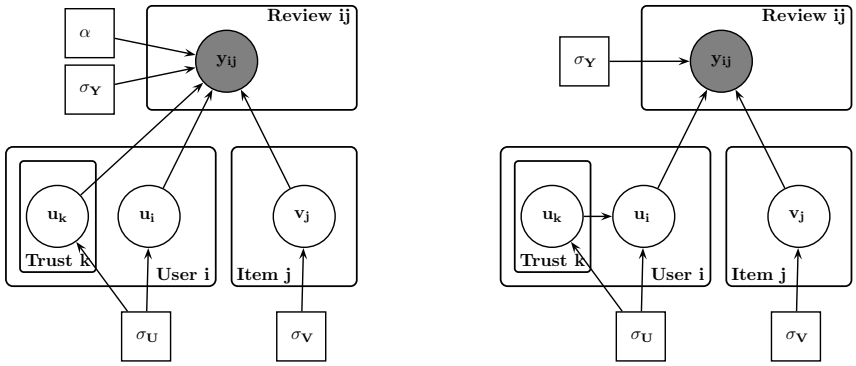


Figure 3.6: Social Recommender

The model showed significant improvement over the PMF models that does not use network information on the experimental dataset. Specially for cold start users, the model showed a prominent improvement of $\sim 40\%$ over PMF.

3.3.2 Recommendations using Social Trust Ensemble (STE)

The main disadvantage of previously described SoRec is that although the users social network is integrated into the CF system by sharing of user latent factors, the real world recommendation processes of a user being influenced by the recommendations of the neighboring friends is not reflected directly in the model. This lends the model uninterpretable and also results in suboptimal use of the network information. The authors of SoRec, later proposed a trust based matrix factorization technique called "Social Trust Ensemble" [79]. In this model, the rating y_{ij} is designed to be a result of combined effects of the user taste and recommendation from the trusted "friends". The user tastes are derived by factorizing the user-item rating matrix as in the case of PMF. To incorporate the effect of recommendations from friends a component of the user propensity is modeled as being derived from the weighted combination of ratings from the members of the user i's trusted friends, N_i, weighted by the normalized confidence values. The ensemble parameter $\alpha \in [0, 1]$, defines the influence of each of these components in determining the user propensity.



(a) STE (b) SocialMF

Figure 3.7: Trust Based CF

For every edge $e_{ij} \in \mathcal{E}$, the normalized trust is defined as $\hat{c}_{ik} = \frac{c_{ik}}{\sum_{k' \in N_i} c_{ik'}}$. The ratings are now modeled as:

$$P(Y|U,V,C,\sigma_Y^2,\alpha) = \prod_{i=1}^{M}\prod_{j=1}^{N}[\mathcal{N}((\alpha \boldsymbol{u_i}^T \boldsymbol{v_j} + (1-\alpha)\sum_{k \in N_i} \hat{c}_{ik} \boldsymbol{u_k}^T \boldsymbol{v_j}), \sigma_Y^2)]^{I_{ij}} \tag{3.24}$$

The latent factors U and V are given spherical Gaussian priors as in PMF. The graphical model is represented in Figure 3.7a. The latent factors are learned by maximizing the posterior log likelihood using stochastic gradient descent. The experimental results showed significant improvement over SoRec which in turn outperforms basic PMF models. Moreover, the computational complexity analysis shows linear scaling as was in the case of SoRec.

3.3.3 Social Matrix Factorization (SocialMF)

An approach similar in flavor to that of STE was proposed by Jamali et. al. in "SocialMF" [80]. In STE, the feature vectors of the trusted friends of a user i affect the rating prediction for the user. In this framework, a users' ratings are influenced only by 1-hop neighbors. This model does not handle trust propagation through the network. In the

SocialMF model, the ratings are influenced only through the user and item latent factors as in PMF. But, unlike PMF, the user latent factors of a user are influenced by the user factors of the trusted friends. The idea is that users with similar tastes form close ties in a social network. Moreover, the latent features of users are indirectly influenced by non-neighboring nodes in the social network, leading to propagation of trust. The model for the ratings is the same as in PMF, given by:

$$P(Y|U,V,\sigma_Y) = \prod_{i=1}^{M}\prod_{j=1}^{N}[\mathcal{N}(y_{ij}|\boldsymbol{u_i}^T\boldsymbol{v_j},\sigma_Y^2)]^{I_{ij}} \tag{3.25}$$

The main difference is in the prior for the user factors. The user and item factor priors are given by:

$$P(U|C,\sigma_U^2,\sigma_C^2) = \prod_{i=1}^{M}\mathcal{N}(\boldsymbol{u_i}|0,\sigma_U^2\mathbb{I})\mathcal{N}(u_i|\sum_{k\in N_i}\hat{c_{ik}}\boldsymbol{u_k},\sigma_C^2\mathbb{I}) \tag{3.26}$$

$$P(V|\sigma_V^2) = \prod_{j=1}^{N}\mathcal{N}(\boldsymbol{v_j}|0,\sigma_V^2) \tag{3.27}$$

The graphical model is given in Figure 3.7b. Again, the latent factors are learned by maximizing the posterior log likelihood through SGD. The results showed improvement over the previously existing state-of-art STE approach strengthening the claim that accounting for transitivity of trust enhances performance of recommendation systems.

3.3.4 Kernelized Probability Matrix Factorization (KPMF)

The main shortcoming of the work discussed so far is that, these models require accurate estimation of the trust in the entire network. Such information in practical scenario is seldom available and the trust estimates based on heuristics are prone to high noise. The noisy information may even lead to degradation of the performance.

In the most recent work by Zhou et. al. [81], the authors propose a generalized approach for incorporating side information from graphs (and other sources of side information) through two zero mean Gaussian process(GP) priors on the latent factors of all users and items. The covariance functions for the GPs are derived from various graph based kernels of the underlying social network. The framework can be extended to include side information from other sources through appropriate choice of kernels. Such a prior captures the correlations between the entity latent factors based on the side information provided. The covariance function of the GP for user and item latent factors are denoted by K_U and K_V respectively. In the paper, the authors derive K_U and K_V from the underlying user social network G. Further they consider an undirected unweighted graph with symmetric adjacency matrix whose entries are 1 if an edge exists between the users and 0 otherwise.

The authors use three graph kernels to capture the correlation among the users in the graph. The Laplacian of the graph is denoted by $L = D - A$, where D is a diagonal matrix with $d_{ii} = \sum_{k=1}^{M} c_{ik}$. *Diffusion kernel* [82], measures the amount of diffusion of influence between nodes along the edges in the graph. This is given by $K_D = e^{-\beta L}$ for some value of the parameter β. *Commute Time kernel (CT)* [72], (also described in [70]), quantifies the average number of steps a random walker takes to commute between the nodes in a graph. As previously stated, this quantity is also related to the Euclidean distance between nodes in the graph and is given by the Moore Penrose pseudo inverse of L, $K_{CT} = L^{+}$. Finally, the *Regularized Laplacian kernel (RL)*, given by $K_{RL} = (1 + \gamma L)^{-1}$, penalizes the variation between adjacent nodes in the graph. These graph based kernels from user social network are used as the covariance functions K_U in the Gaussian process.

The generative process for the ratings is similar to that of PMF given by Equa-

tion 2.3 while the priors on the user and item latent factors are as given below:

$$P(U|K_U) = \prod_{d=1}^{D} \mathcal{N}(U_{:,d}|\mathbf{0}, K_U) \tag{3.28}$$

$$P(V|K_V) = \prod_{d=1}^{D} \mathcal{N}(V_{:,d}|\mathbf{0}, K_V) \tag{3.29}$$

where $U_{:,d} \in \mathbb{R}^M$ and $V_{:,d} \in \mathbb{R}^N$ represent the dth column vectors of matrices U and V respectively, which are also the vectors of the dth latent variable across all users and items respectively. Maximizing the log-posterior of U and V over the observed R (ignoring the constants) leads to the following optimization problem which can be solved using gradient descent techniques.

$$U, V = \underset{U,V}{argmin} \frac{1}{2\sigma_Y^2} \sum_{i=1}^{M} \sum_{j=1}^{N} I_{ij}(y_{ij} - \mathbf{u_i}^T \mathbf{v_j})^2 + \frac{1}{2} \sum_{d=1}^{D} (U_{:,d}^T K_U^{-1} U_{:,d} + V_{:,d}^T K_V^{-1} V_{:,d}) \tag{3.30}$$

In effect the regularizations on the user and item factors are influenced by the available external side information. The experimental evaluation shows considerable improvement over the basic PMF models, especially for sparse training data. Moreover, the results were shown to outperform the previously described SoRec system [78] which also uses social network as side information in PMF. Further, the CT kernel was shown to give the best performance on the datasets experimented with. Finally, the system showed capability for handling cold start users in the presence of richly connected neighbors for the user.

3.4 Observation Sensitivity

Existing approaches to affinity estimation assume that the missing affinities are missing at random (MAR). However, this assumption is generally incorrect as is evident from a large number of affinity expressing datasets that are highly sparse with highly non-random distribution of observed affinities. A small fraction of entities account for a large

fraction of the available data while the remaining observations are sparsely distributed among the remaining entities. This kind of a distribution is commonly known as "power-law" [83]. Moreover, the ratings themselves are not uniformly distributed across the values. In a typical rating system, a user often chooses to rate a particular item if there is a strong positive or negative affinity for that item. Thus, the number of neutral values observed is low. Ignoring this dependence can result in biased models.

The publications by Marlin et. al. [84, 85], have analyzed the effects of MAR assumptions on the predictive accuracy of recommendation systems. Let R represents the indicator matrix of observing the entries of the response matrix Y. Also, let Z represent the set of latent variables, (U, V in a basic PMF model) and Θ represent the model parameters for modeling of the response matrix Y. Finally, let μ represent the parameter controlling the distribution of R. The joint distribution of R, Y and Z can be factorized as follows:

$$P(R, Y, Z|\Theta, \mu) = P(R|Y, Z, \mu)P(Y, Z|\Theta) \tag{3.31}$$

The first term $P(R|Y, Z, \mu)$ is referred as the "missing data model" and the second term $P(Y, Z|\Theta)$ is called the "data model". The above factorization is derived from the intuition that data is generated from an underlying *data model* and based on the *missing data model*, some of the entries are observed.

The missing at random assumpyion believes that, given the observed data, the observation matrix R is independent on unobserved data. Mathematically, this implies that $P(R|Y_\kappa, Y_{\kappa^c}, Z, \mu) = P(R|Y_\kappa, \mu)$. When the data is missing at random, maximum likelihood inference based on the observed data only is unbiased. This can be verified my marginalizing $P(R, Y, Z|\Theta, \mu)$ w.r.t Z and Y_{κ^c}. However, when data is not missing at random, this important property fails to hold, and ML approaches lead to biased estimates.

In the former work, [84], the authors study the impact of MAR assumption on the predictive performance of collaborative filtering using synthetic data experiments. In the later work, [85], the authors demonstrate the same through real life user study in association with Yahoo!Lauchcast. They first show the ratings generated from users deviate significantly from the MAR assumption. In a separate survey, the response of a significant 35% of the users claimed to avoid rating items for which they have a neutral opinion. These analysis, strengthen the intuition against MAR asssumption on CF. These papers also develop heuristic models to incorporate data generation procedure into the *multinomial mixture model* [13] framework. However, their extensions to more succesful matrix factorization models has not been full formulated.

The chapter is concluded with a brief mention of similar work using side-information for response prediction. In binary link and *click-through-rate* predictions, Menon et. al.[86] develop a set of heuristics for *click through rate* prediction by incorporating hierarchies and side information. In "Localized Factor Models" developed in [87], and the references therein, a framework for multi context recommendation with provisions to incorporate side information as one of the context is developed. In the work by Adams. et. al. [88], the authors propose a novel method for incorporating side information through Gaussian processes. The model known as "Dependent Probabilistic Matrix Factorization (DPMF)" incorporates the side information by coupling together multiple PMF models spanning the feature space of side information, through a Gaussian process priors. The latent factors in PMF models are now replaced with functions that vary over the covariate/temporal space. A Gaussian process prior is now placed on the latent functions which mandates the function to be smooth in covariate space and share information. This model can use both temporal

and/or entity attributes.

Though, extensive work, has been completed towards side information aware recommendation systems, there are multiple avenues for improvement in terms of interpretability, scalability, and ease for online update of the system.

Chapter 4

Review Quality Aware Collaborative Filtering

Probabilistic matrix factorization (PMF) and other popular approaches to collaborative filtering discussed so far assume that the ratings given by users for products are genuine, and hence equal importance is given to all available ratings. However, this is not always true due to several reasons including the presence of opinion spam in product reviews. Often, users unhappy with a seller tend to give a poor rating for the product, which does not necessarily reflect on the quality of the product. Other times, some sellers might deliberately give superior ratings to promote their products, or they might give unjust poor ratings to competitors' products. The presence of such spurious ratings could impact the performance of the underlying collaborative filtering model [89]. In this chapter, the possibility of performing collaborative filtering while attaching weights or quality scores to the ratings is explored. The quality scores, which are determined from the corresponding review data are used to "up–weight" or "down–weight" the importance given to the individual ratings while performing collaborative filtering, thereby improving the accuracy of the predictions.

On one hand, there is a large body of work on the analysis of online product reviews, especially in the area of assessing the helpfulness of online reviews [90, 91]. Kim et al. [90] propose a quantitative measure based on the review feedback information to assess the helpfulness of reviews. A regression model using various features extracted from the review text is trained to predict the helpfulness score for new reviews. O'Mahony and Smyth [92]

model the same problem as a classification task. Rather than predicting a score for helpfulness, a classifier is trained using reputation, content, social, and sentiment based features derived from user and item metadata to classify a review as helpful or unhelpful. Danescu-Niculescu-Mizil et al. [91] study the correlation of different aspects of review metadata with review helpfulness. The results of the study show that there is a strong correlation between the signed deviation of the review rating to the average rating of the product. There are several approaches proposed in the literature for opinion spam detection [93–96]. Jindal and Liu [93] train a classifier based on user, item, and review metadata to identify different categories of spam in online reviews. Liu et al. [94] propose a method to detect low quality reviews in order to improve the quality of opinion summarization. Ott et al. [95] propose approaches to detect fictitious and imaginative opinions that have been deliberately written to sound authentic. Further, O'Mahony et al. [97] examine the robustness of various collaborative recommendation techniques in the face of malicious attacks. Theoretical results on recommendation accuracy and stability in the presence of malicious agents is derived. Mobasher et. al [89] analyze various new attack models and their impact on recommendation algorithms through extensive simulation-based evaluation. In a more recent work, Wu et al. [98] propose a semi–supervised learning algorithm to identify spam reviews/shillings using user metadata. The spam reviews are then removed from the training set while performing collaborative filtering. However, none of these approaches provide a robust methodology to improve the performance of the recommendation systems in the presence of opinion spam.

On the other hand, there is also a fair amount of work in the area of collaborative filtering for recommender systems that has been discussed in the previous chapters. The model proposed here, tries to combine online product review helpfulness with collaborative

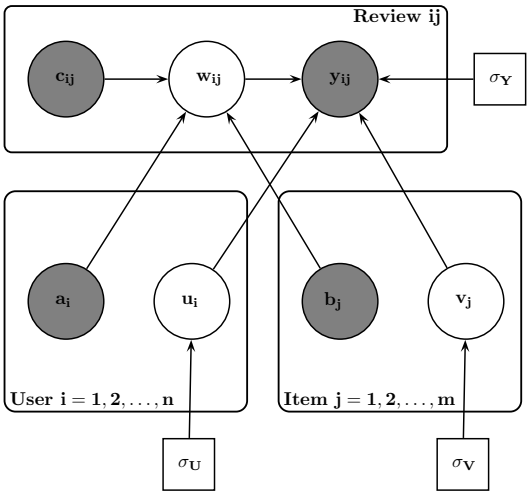


Figure 4.1: Graphical model for the two stage approach to recommender systems

filtering to improve the overall performance of recommender systems.

4.1 Model description

The model consists of two stages. In *Stage 1*, the quality scores of ratings are estimated using the review and user data. In *Stage 2*, these quality scores are used as weights assigned to ratings and *weighted probabilistic matrix factorization* is performed on the ratings to predict new recommendations.

The following additional notation is used in this chapter. A quality score, w_{ij} is associated with each rating, y_{ij}. The user and item meta data and review based features, which are used in the estimation of quality scores are represented by $\mathbf{a_i}$, $\mathbf{b_j}$ and $\mathbf{c_{ij}}$ respectively. The graphical model for the approach is given in Figure 4.1.

4.1.1 Stage 1: Quality Score Estimation

The quality score for a rating is reflective of the authenticity of the rating. Most websites like Amazon and Yelp allow users to indicate if reviews were helpful or not. The amount of positive feedback obtained by a product review is indicative of the authenticity of the corresponding rating. Kim et al. [90] have proposed the measure below to quantify helpfulness for online product reviews based on the feedback information.

$$\text{helpfulness} = \frac{\text{Number of helpful votes}}{\text{Total number of votes}} \tag{4.1}$$

The score computed as described above is a fair indication of quality if the amount of feedback is sufficiently high. However, for more recent reviews that have low feedback, this score might not necessarily capture the true quality of the rating. For such reviews, the quality score is estimated using a regression model trained on those reviews that have sufficient feedback. The quality score computed using the formula described above is used as the response variable in the regression model. Several features are used for training the regression model:

- **Metadata based features**

  Along with the natural language text, most reviews are associated with side information about the users and reviews, which are used as features to train the regression model. Specifically, features like the average rating given to the user that indicates how useful his reviews have been, duration for which the review has been around, deviation of the rating from the mean rating of the product, length of the title of the review, and length of the review text are used in this approach. Note that item

based metadata information was not used as features in the regression model since item related features do not necessarily indicate the quality of the review.

- **Text based features**

  In most online websites, ratings are accompanied by reviews written by the users in natural language text. Several different features are extracted from the review text for training the regression model:

  1. *Unigram counts or bag-of-words features*

     Kim et al. [90] have demonstrated that the unigram counts or bag-of-words based features have been very useful for predicting the helpfulness score. Thus, bag-of-words features are extracted from the review text to train the regression model.

  2. *Features from topic modeling*

     Certain words in the review text like *good*, *bad*, *colorful*, are better indicators of quality than remaining words. One approach to discovering these *latent topics* in natural language text involves using techniques from topic modeling like Latent Dirichlet Allocation (LDA) [99]. LDA is applied to discover latent topics in the review text. The latent topic probabilities are then used as features to train the regression model.

In Stage 2, the quality scores estimated in Stage 1 are used to build a recommendation system based on collaborative filtering. Among the methods used for collaborative filtering, latent factor based matrix factorization models have been shown to give the best performance in most scenarios. Thus, a basic probabilistic matrix factorization framework

[8] is adapted to incorporate quality scores. This method can be extended to other matrix factorization based models. The quality score is modeled as a factor that inversely affects the variance of the affinity response from the mean of the factor model. The intuition is that higher quality ratings are given a prior with lower deviations from the model and thus their deviations from the model mean are more heavily penalized. On the other hand, low quality scores are allowed larger deviations from the model mean. The priors for U and V, are as in Equation 2.4. The modified prior on Y is given by:

$$P(Y|U,V) \;=\; \prod_{i=1}^{M}\prod_{j=1}^{N}\left[\mathcal{N}\left(\mathbf{u_i}^T\mathbf{v_j}, \frac{\sigma_Y^2}{w_{ij}}\right)\right]^{I_{ij}} \tag{4.2}$$

Maximizing log posterior of observed Y_κ in this model leads to the minimization objective given below, which follows an intuitive interpretation of minimizing the weighted squared error of the observed ratings, regularized appropriately:

$$L(\theta) \;=\; \sum_{i,j} I_{ij}[w_{ij}(y_{ij} - \mathbf{u_i}^T\mathbf{v_j})^2] + \lambda_1\|U\|_F^2 + \lambda_2\|V\|_F^2 \tag{4.3}$$

where $\lambda_1 = \sigma_Y^2/\sigma_U^2$ and $\lambda_2 = \sigma_Y^2/\sigma_V^2$. In the experiments, it is assumed that, $\lambda_1 = \lambda_2 = \lambda$. The above objective is a differentiable function in $\mathbf{u_i}$ and $\mathbf{v_j}$ and the maximization can be performed using the Stochastic Gradient Descent (SGD) algorithm.

4.2 Experiments

The efficacy of the approach is demonstrated through experiments on two product categories of a benchmark data set from Amazon.com. The open source data set provided by Jindal and Liu [93] was used in these experiments. The categories of *Books* and *Audio CDs* were used for experimental evaluation as they had a reasonable number of users, products, and reviews. Multiple reviews for a single user-item pair were eliminated and the latest

	Books	**Audio CDs**
Total Users	772674	46491
Total Items	493991	29477
No. Ratings in Training	1677892	75518
No. Ratings in Validation	41081	902
No. Ratings in Test	105285	2683

Table 4.1: Various statistics about the data sets used in experimental evaluation.

reviews based on time stamp was retained. The set of available ratings was split into training, validation and test sets for the recommendation system based on time. The reviews in validation appeared after the ones in training and the reviews in test were posted later than those in validation. Table 4.1 gives details about the two data sets used in the experimental evaluation.

Regression models were trained for predicting quality scores using three different sets of features – text, metadata and text+metadata. For the Books data set, all those reviews that had feedback from more than 50 users were used for training the regression model. Since the Audio CDs data set had reviews with fewer users providing feedback, reviews that had feedback from more than 20 users were used for training the regression model.

The SRI Language Modeling Toolkit[1] was used to extract unigram counts for bag-of-words features. Commonly occurring stop words were removed while extracting bag-of-words features. Unigram counts were normalized to obtain term frequency values, which were then used as features in the regression model. There were around 760,000 unique words in total in the Books data set and around 128,000 unique words in the Audio CDs data set. Following the Zipf's law, a large number of these occurred very rarely in the entire corpus.

[1] http://www.speech.sri.com/projects/srilm

Words that occurred less than 10 times in the corpus were eliminated. David Blei's LDA implementation[2] was used to identify latent topics in the review text. With the reduced set of words, LDA was scalable on the large number of reviews dealt with. LDA was run with 5, 10, and 50 latent topics and the latent topic probabilities were used as features to train the regression model. The resulting models are called LDA_5, LDA_{10}, and LDA_{50} respectively. Finally, the raw feature values from metadata were scaled to a value between 0 and 1 by the transformation suggested in [100], $\tilde{f} = \frac{\log(f+1)}{1+\log(f+1)}$, where f is the original value of the feature.

Regression models were trained using the techniques described below:

- **Logistic Regression**

  Since the quality scores lie in the range 0 to 1, the choice of logistic regression was suitable as the predicted values from the logistic regression model lie between 0 and 1. Logistic regression models were trained by specifying the response in terms of the number of users that provided positive feedbacks and total number of users that provided feedback using features extracted from LDA and metadata.

- **Support Vector Regression**

  Support Vector Regression (SVR) [101] was used for bag-of-words features extracted from natural language text since it can handle any number of features and feature vectors of arbitrary size. One issue with SVR is that it does not guarantee to predict a value between 0 and 1 for the test example. To overcome this, quality scores of the reviews were mapped to real values in $\mathbb{R}$ using the inverse logistic function. The

[2]http://www.cs.princeton.edu/ blei/lda-c/

mapped scores were used as response variables to train the SVR model. The predicted value for a test sample was then passed through the logistic function to get the quality score. The linear kernel was used since it performed the best in the preliminary experiments. Note that SVR was used only when feature vectors were not of fixed size, like in bag-of-words, since the implementations of logistic regression and LASSO used were inefficient for large feature vectors due to the lack of support for sparse vector representation.

- **LASSO Regression**

  For features extracted from metadata and LDA, a regression model was trained using lasso regression since it helps identify more useful features from the entire set. Like in SVR, quality scores of the reviews were mapped to real values in $\mathbb{R}$ using the inverse logistic function, and the predicted scores were later mapped back to get a value between 0 and 1. The python interface for LASSO in Sklearn package [102] was used in the experimental evaluation.

For the baseline estimate, the default implementation of PMF in Graphlab [103], that uses alternating least squares method to perform factorization was used. The number of latent factors was set to $D = 40$. Grid search was performed to tune the regularization parameter λ using the validation data set over the range of 0.01 to 0.8. For the first baseline model, referred to as "vanilla PMF", probabilistic matrix factorization was performed by setting weights for all reviews to 1. For the second baseline, "second baseline", quality scores were estimated from the feedback votes for those reviews that had sufficient feedback (50 or more for Books and 20 or more for Audio CDs). For the remaining reviews, the weights were set to the average of the scores that were estimated using feedback information in

the previous step and weighted probabilistic matrix factorization was performed. Initial experiments with these two models showed marginal improvement in the performance of latter over former. This observation supported the hypothesis that incorporating quality scores improves the performance of the recommender system.

4.3 Results and Discussion

10-fold cross validation was performed on the training set and root mean square error (RMSE) was computed to measure the performance of the regression models. These quality scores were used in the weighted PMF model previously described. Table 4.2 shows the RMSE for different models of collaborative filtering on the test set for both Books and Audio CDs. On the Books data set, all models including the second baseline outperform the vanilla PMF, while on Audio CDs data set, a majority of the models outperform the vanilla PMF. Lack of sufficient data in the Audio CDs data set could possibly be the reason for inferior performance of some models.

Logistic regression model trained on metadata features is the best performing model on the Books data set and it results in an improvement of 0.0355 (2.49%) over vanilla PMF and 0.0344 (2.41%) over the second baseline. However, on the Audio CDs data set, SVR trained on metadata and bag-words features is the best performing model, with a performance improvement of 0.0175 (1.27%) over vanilla PMF and 0.0128 (.93%) over the second baseline. In general, models trained only on metadata based features perform better than those trained on both LDA and metadata based features indicating strong signals from the metadata features used – time stamp, length of review text, length of review title, rank of the user and deviation of the rating from the mean rating of the product. Even though

Model	Books	Audio CDs
Vanilla PMF	1.4230 ($\lambda = 0.35$)	1.3739 ($\lambda = 0.30$)
Second Baseline	1.4219 ($\lambda = 0.35$)	1.3692 ($\lambda = 0.20$)
LR-metadata	**1.3875** ($\lambda = 0.25$)	1.3664 ($\lambda = 0.25$)
LR-metadata+LDA_5	1.3972 ($\lambda = 0.20$)	1.3740 ($\lambda = 0.25$)
LR-metadata+LDA_{10}	1.3966 ($\lambda = 0.25$)	1.3779 ($\lambda = 0.25$)
LR-metadata+LDA_{50}	-	1.3731 ($\lambda = 0.25$)
LASSO-metadata	1.3910 ($\lambda = 0.30$)	1.3662 ($\lambda = 0.30$)
LASSO-metadata+LDA_5	1.3952 ($\lambda = 0.30$)	1.3634 ($\lambda = 0.30$)
LASSO-metadata+LDA_{10}	1.3958 ($\lambda = 0.30$)	1.3745 ($\lambda = 0.20$)
LASSO-metadata+LDA_{50}	-	1.3680 ($\lambda = 0.30$)
SVR-metadata+bag-of-words	1.4135 ($\lambda = 0.30$)	**1.3564** ($\lambda = 0.30$)
SVR-bag-of-words	1.4219 ($\lambda = 0.30$)	1.3740 ($\lambda = 0.30$)
Best-Model-Low-Quality-Scores-Dropped	1.3945 ($\lambda = 0.25$)	**1.3389** ($\lambda = 0.30$)

Table 4.2: Test RMSE for Books and Audio CDs data sets in Stage 2. "LR", "LASSO", and "SVR" refer to the models in which quality scores are predicted using logistic regression, LASSO regression, and support vector regression respectively.

models trained with text based features are outperformed by the metadata based models, they still show significant improvement over the baseline models on both data sets, which warrants further investigation into linguistic feature engineering on review text. Overall, the results indicate that incorporating quality scores as weights for ratings in collaborative filtering improves the performance of recommender systems.

Finally, the impact of using ratings with low quality scores in training the PMF was studied. The hypothesis was that ratings with poor quality scores could possibly affect the predictions adversely, and hence eliminating them during training might further improve the performance of recommender systems. Figures 4.2 shows the distribution of quality scores from LR-metadata, which is the best performing model on the Books data set. While most of ratings have reasonably high quality scores, a small number of them have fairly poor quality scores. Analysis of the distribution of quality scores from SVR-metadata+bag-of-

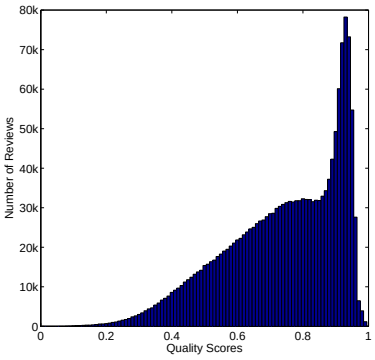


Figure 4.2: Distribution of quality scores from LR-metadata, the best performing model on the Books data set.

words, the best performing model on the Audio CDs data set yielded similar results. In this experiment, all ratings with a quality score less than 0.4 were eliminated and PMF was performed with the remaining ratings using the best performing model on both data sets. The results from these experiments, "Best-Model-Low-Quality-Scores-Dropped" are shown in Table 4.2. Eliminating low quality scores improved the results on the Audio CDs data set considerably, thereby supporting the hypothesis. However, the performance on the Books data set dropped marginally.

4.3.1 Quality Indicators

Regression models were analyzed to identify features that impacted the quality of the rating. First, coefficients learned from LASSO and logistic regression on metadata features were examined on the two data sets. Both the regression models had learned similar coefficients for individual metadata features. On both data sets, the review length had the highest positive coefficient, while the deviation of the ratings from the mean rating had the highest negative coefficient from both models. These observations follow from the

intuition that longer reviews are indicators of a thorough analysis of the product by the reviewer and hence the reviewer's rating is highly reliable. On the other hand, a reviewer giving a rating that is highly deviant from the mean rating is likely to be a spammer with a malicious intention of either boosting or degrading a product popularity and hence the negative correlation with the quality. The other features like time stamp and review title length were found to be not very influential in estimating the quality as both regression models assigned low or near-zero coefficients to these features.

Next, regression coefficients learned using LASSO on topics induced by LDA were examined. There were two topics induced by LDA_5 that had high negative coefficients. Some of the words from the former topic included *information*, *good*, *great*, *excellent*, *guide*, *books*, while the words from the latter topic included *history*, *book*, *war*, *world*, *people*, *american*. The remaining topics had low or near zero coefficients indicating that they did not play a significant role in determining the quality of the rating. While the words in the former topic indicate strong opinions which could be used to mask the real quality of the products, the words in the latter topic mostly describe different categories of books, which might not necessarily describe the quality of the product. In general, it was found that LDA was more inclined to clustering thematic topics together rather than topics that were indicative of quality. Its inability to distinguish thematic words from quality indicators is possibly one of the reasons for the modest performance of LDA-based features in the experiments. The analysis of topics induced by LDA_{10} yielded similar results on the Books data set. Further analysis of words induced by LDA on the Audio CDs data set did not yield any interesting observations. Overall, these results emphasize the need for extraction of more sophisticated features from the review text.

In summary, incorporating quality scores or weights to ratings improves the performance of collaborative filtering in recommender systems. Experiments with different types of features extracted from both review metadata and text indicate that some of the metadata-based features are highly indicative of the quality of the rating. Further, experiments with text-based features also demonstrate promise, but also indicate the need for extraction of more sophisticated features from review text. Overall, the two stage approach to collaborative filtering is a robust method that is capable of overcoming the negative effects caused by spurious reviews and ratings in recommender systems.

Chapter 5

Mixed Membership Bayesian Affinity Estimation

This chapter, introduces a novel *Mixed Membership Bayesian Affinity Estimation (MMBAE)* model towards a side information aware affinity estimator. The model efficiently exploits the available entity attributes information within a common framework to predict affinity relationships. This model is inspired from "Simultaneous decomposition and Prediction (SDaP)" framework, that iteratively partitions a heterogeneous prediction task into homogeneous and manageable pieces while concurrently building multiple predictive models, one for each piece. The joint feature and model selection framework results in more interpretable and accurate models. Further, the model allows for mixed cluster membership for each user and item to accurately capture the inherent mixed membership effects of the recommendation systems as discussed in [40, 43, 45]. Finally, a Bayesian approach is adopted for the problem that allows for imputation of missing attributes and avoids overfitting by point estimates.

First the generative process for the model is described. The inference algorithm based on variational approximation methods is then derived. Finally, preliminary experiments and their results are discussed.

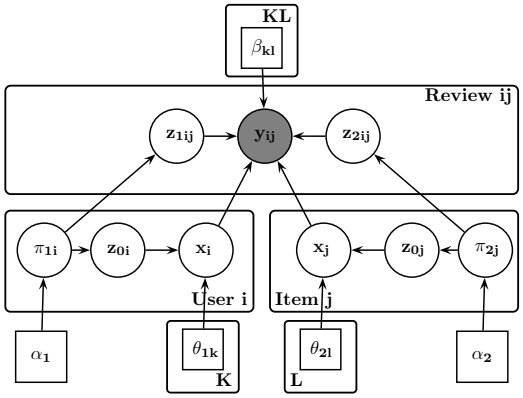


Figure 5.1: Mixed Membership Bayesian Affinity Estimation (MMBAE)

5.1 Model Description

The graphical model depicting generative process for the observed entities and the ratings is given in Figure 5.1. For each user i and item j, the mixing coefficients, $\boldsymbol{\pi_{1i}} \in \mathbb{R}^K$ and $\boldsymbol{\pi_{2j}} \in \mathbb{R}^L$ are sampled from Dirichlet distributions $Dir(\boldsymbol{\alpha_1})$ and $Dir(\boldsymbol{\alpha_2})$ respectively. For each observed rating y_{ij}, a co-cluster $z_{1ij}z_{2ij}$ is sampled from the user and item cluster membership distributions. The rating value is modeled through a co-cluster specific, generalized linear model (GLM) [104] over the user attributes, $\boldsymbol{x_{1i}}$ and the item attributes, $\boldsymbol{x_{2j}}$. The co-cluster specific GLM parameters for the co-cluster, (kl), are denoted by β_{kl}. Further, a separate user and item cluster assignments, z_{0i} and z_{0j}, are sampled to generate the entity attributes $\boldsymbol{x_{1i}} \in \mathbb{R}^{D1}$ and $\boldsymbol{x_{2j}} \in \mathbb{R}^{D2}$ respectively. The attribute values are generated from a member of exponential family of distributions whose parameters are specified by the cluster assignments z_{0i} and z_{0j} respectively. These distribution parameters for user cluster k and item cluster l are denoted by θ_{1k} and θ_{2l} respectively. The use of exponential family of distributions for modeling the attributes, provides great flexibility in modeling diverse data types within a single framework. The generative process for the model is described in

Model 6.

Model 6 Generative model for MMBAE

for Users, $i = 1, 2, \ldots, M$ **do**
  Sample mixing coefficients, $\boldsymbol{\pi_{1i}} \sim Dir(\boldsymbol{\alpha_1})$
  Sample attribute generating cluster assignment, $z_{0i} \sim Mult(\boldsymbol{\pi_{1i}})$
  Sample attributes, $\boldsymbol{x_{1i}} \sim P_{\psi_1}(\boldsymbol{x_{1i}}|\theta_{1z_{0i}})$
end for
for Items, $j = 1, 2, \ldots, N$ **do**
  Sample mixing coefficients, $\boldsymbol{\pi_{2j}} \sim Dir(\boldsymbol{\alpha_2})$
  Sample attribute generating cluster assignment, $z_{0j} \sim Mult(\boldsymbol{\pi_{2j}})$
  Sample attributes, $\boldsymbol{x_{2i}} \sim P_{\psi_2}(\boldsymbol{x_{2j}}|\theta_{2z_{0j}})$
end for
for Ratings $y_{ij} \in Y_\kappa$ **do**
  Sample user cluster membership, $z_{1ij} \sim Mult(\boldsymbol{\pi_{1i}})$
  Sample item cluster membership, $z_{2ij} \sim Mult(\boldsymbol{\pi_{2j}})$
  Sample the rating values, $y_{ij} \sim P_{\psi_Y}(y_{ij}|\boldsymbol{\beta_{z_{1ij}z_{2ij}}}^T[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}])$
end for

The set of user and item mixing coefficients are denoted by a $M \times K$ matrix, $\Pi_1 = \{\pi_{1i}\}$ and a $N \times L$ matrix, $\Pi_2 = \{\pi_{2j}\}$ respectively. The attribute generating cluster assignments for users and items are denoted by $Z_{01} = \{z_{0i}\}$ and $Z_{02} = \{z_{0j}\}$ respectively. The user and item cluster memberships for the observed ratings in κ are denoted by $Z_1 = \{z_{1ij}\} \in \mathbb{R}^{\kappa \times 1}$ and $Z_2 = \{z_{2ij}\} \in \mathbb{R}^{\kappa \times 1}$ respectively. The user and item attributes are collectively represented by matrices X_1 and X_2 respectively, where the rows of the matrices represent the attributes for each entity. The user and item distribution parameters are given by $\Theta_1 = \{\theta_{1k}\} \in \mathbb{R}^{M \times K}$ and $\Theta_2 = \{\theta_{2l}\} \in \mathbb{R}^{N \times L}$ respectively. Finally, the ratings distribution parameters are given by $\boldsymbol{\beta} = \{\beta_{kl}\} \in \mathbb{R}^{K \times L \times (D1+D2)}$.

The overall joint distribution over all observed and latent variables is given by:

$$
\begin{aligned}
&P(Y_\kappa, X_1, X_2, Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2 | \boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta}) = \\
&\prod_{i=1}^{M} P(\boldsymbol{\pi_{1i}}|\boldsymbol{\alpha_1}) P(z_{0i}|\boldsymbol{\pi_{1i}}) P_{\psi_1}(\boldsymbol{x_{1i}}|\theta_{1i}) \\
&\prod_{j=1}^{N} P(\boldsymbol{\pi_{2j}}|\boldsymbol{\alpha_2}) P(z_{0j}|\boldsymbol{\pi_{2j}}) P_{\psi_2}(\boldsymbol{x_{2j}}|\theta_{2j}) \\
&\prod_{(ij)\in\kappa} P(z_{1ij}|\boldsymbol{\pi_{1i}}) P(z_{2ij}|\boldsymbol{\pi_{2j}}) P_{\psi_Y}(y_{ij}|\beta^T_{z_{1ij}z_{2ij}}[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}])
\end{aligned}
\tag{5.1}
$$

5.2 Inference

The model parameters $(\boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta})$, can be learned by maximizing the observed log likelihood obtained by marginalizing the latent variables in Equation 5.1. However, computation of the observed log likelihood in exact form is intractable for the model. Thus, variational mean field approximation is used for inference [20]. The mean field approximation assumes a factorized form of the posterior of the latent variables and tractable lower bound on the observed likelihood is obtained which is in turn maximized with respect to the model parameters. The true posterior $P(Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2 | Y_\kappa, X_1, X_2, \boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta})$ is approximated by the following factorized form:

$$
\begin{aligned}
q(Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2) \quad = &\prod_{i=1}^{M} q(\boldsymbol{\pi_{1i}}|\boldsymbol{\gamma_{1i}}]) q(z_{0i}|\boldsymbol{r_{0i}}) \prod_{j=1}^{N} q(\boldsymbol{\pi_{2j}}|\boldsymbol{\gamma_{2j}}) q(z_{0j}|\boldsymbol{r_{0j}}) \\
&\prod_{(ij)\in\kappa} q(z_{1ij}|\boldsymbol{r_{1ij}}) q(z_{2ij}|\boldsymbol{r_{2ij}})
\end{aligned}
\tag{5.2}
$$

Further, let Q be the set of all distributions over the latent variables, which are of the form in Equation 5.2. Using Jensens' inequality, the following lower bound on the observed log likelihood is derived:

$$
\begin{aligned}
&\log P(Y_\kappa, X_1, X_2 | \boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta}) \geq L(q) \\
&L(q) = H(q) + \mathbb{E}_q[\log P(Y_\kappa, X_1, X_2, Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2 | \boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta})]
\end{aligned}
\tag{5.3}
$$

where H(q) is the entropy of the variational distribution $q \in Q$ and $\mathbb{E}_q[(.)]$ is the expectation with respect to the variational distribution. An Expectation Maximization algorithm is used for parameter estimation.

5.2.1 E-Step

In the Expectation-Step (E-Step), the task to identify $q^* \in Q$, such that,

$$q^* = \underset{q \in Q}{argmax} L(q)$$

Based on the derivation in [105], for the family of Q described above, the maximizing distribution is given by $q^* = \prod_k q_k^*$, where k indexes the latent variables in the factorized family and q_i^* are given by the following equations:

$$q_i^* \propto exp(\mathbb{E}_{q|q_i}[\log P(Y_\kappa, X_1, X_2, Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2 | \boldsymbol{\alpha_1}, \boldsymbol{\alpha_2}, \Theta_1, \Theta_2, \boldsymbol{\beta})]) \tag{5.4}$$

The following equations are derived by evaluating the above distributions:

$$\begin{aligned} q^*(Z_{01}, Z_{02}, Z_1, Z_2, \Pi_1, \Pi_2) \quad &= \prod_{i=1}^{M} q^*(\boldsymbol{\pi_{1i}} | \boldsymbol{\gamma_{1i}}) q(z_{0i} | \boldsymbol{r_{0i}}) \prod_{j=1}^{N} q^*(\boldsymbol{\pi_{2j}} | \boldsymbol{\gamma_{2j}}) q(z_{0j} | \boldsymbol{r_{0j}}) \\ &\quad \prod_{(ij) \in \kappa} q^*(z_{1ij} | \boldsymbol{r_{1ij}}) q^*(z_{2ij} | \boldsymbol{r_{2ij}}) \end{aligned}$$

where,

$$\begin{aligned} q^*(\boldsymbol{\pi_{1i}} | \boldsymbol{\gamma_{1i}}) \quad &= Dir(\boldsymbol{\gamma_{1i}}) \qquad q^*(\boldsymbol{\pi_{2j}} | \boldsymbol{\gamma_{2j}}) \quad = Dir(\boldsymbol{\gamma_{2j}}) \\ q^*(z_{0i} | \boldsymbol{r_{0i}}) \quad &= Mult(\boldsymbol{r_{0i}}) \qquad q^*(z_{0j} | \boldsymbol{r_{0j}}) \quad = Mult(\boldsymbol{r_{0j}}) \\ q^*(z_{1ij} | \boldsymbol{r_{1ij}}) \quad &= Mult(\boldsymbol{r_{1ij}}) \qquad q^*(z_{2ij} | \boldsymbol{r_{2ij}}) \quad = Mult(\boldsymbol{r_{2ij}}) \end{aligned}$$

The variational parameters are given by the following set of equations:

$$
\begin{aligned}
\boldsymbol{\gamma_{1i}} &= \boldsymbol{\alpha_1} + \boldsymbol{r_{0i}} + \sum_{j:(i,j)\in\kappa} \boldsymbol{r_{1ij}} \\
\boldsymbol{\gamma_{2j}} &= \boldsymbol{\alpha_2} + \boldsymbol{r_{0j}} + \sum_{i:(i,j)\in\kappa} \boldsymbol{r_{2ij}} \\
r_{0i(k)} &\propto p_{\psi_1}\left(\boldsymbol{x_{1i}}|\theta_{1k}\right) e^{\psi\left(\gamma_{1i(k)}\right) - \psi\left(\sum_{k'=1}^{K}\gamma_{1i(k')}\right)} \\
r_{0j(k)} &\propto p_{\psi_2}\left(\boldsymbol{x_{2j}}|\theta_{2l}\right) e^{\psi\left(\gamma_{2j(l)}\right) - \psi\left(\sum_{l'=1}^{L}\gamma_{2j(l')}\right)} \\
r_{1ij(k)} &\propto exp\left(\psi\left(\gamma_{1i(k)}\right) - \psi\left(\sum_{k'=1}^{K}\gamma_{1i(k')}\right) + \sum_{l=1}^{L} r_{2ij(l)} \log P_{\psi_Y}\left(y_{ij}|\beta_{kl}^{T}[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}]\right)\right) \\
r_{2ij(l)} &\propto exp\left(\psi\left(\gamma_{2j(l)}\right) - \psi\left(\sum_{l'=1}^{L}\gamma_{2j(l')}\right) + \sum_{k=1}^{K} r_{1ij(k)} \log P_{\psi_Y}\left(y_{ij}|\beta_{kl}^{T}[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}]\right)\right)
\end{aligned}
\tag{5.5}
$$

where $\psi(x) = \frac{d}{dx}\ln\Gamma(x)$ is the Digamma function[1].

5.2.2 M-Step

The parameters of the model are updated with the values which maximize $L(q)$. The maximizing values are given by the following equations:

$$
\begin{aligned}
\theta_{1k} &= \nabla\psi_1^{-1}\left(\frac{\sum_{i=1}^{M} r_{0i(k)}\boldsymbol{x}_{1i}}{\sum_{i=1}^{M} r_{0i(k)}}\right) \\
\theta_{2l} &= \nabla\psi_2^{-1}\left(\frac{\sum_{j=1}^{N} r_{0j(l)}\boldsymbol{x}_{2j}}{\sum_{j=1}^{N} r_{0j(k)}}\right) \\
\beta_{kl} &= \underset{\beta}{argmin} \sum_{(i,j)\in\kappa} r_{1ij(k)} r_{2ij(l)}[\langle y_{ij}, \beta^T[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}]\rangle - \psi_Y(\beta^T[\boldsymbol{x_{1i}}\ \boldsymbol{x_{2j}}])] \\
\alpha_1 &= \underset{\alpha}{argmin} \sum_{i=1}^{M}\left[\log\frac{\Gamma\left(\sum_{k=1}^{K}\alpha_{(k)}\right)}{\prod_{k'=1}^{K}\Gamma(\alpha_{(k')})} + \sum_{k=1}^{K}(\alpha_{(k)}-1)\left[\psi\left(\gamma_{1i(k)}\right) - \psi\left(\sum_{k'=1}^{K}\gamma_{1i(k')}\right)\right]\right] \\
\alpha_2 &= \underset{\alpha}{argmin} \sum_{j=1}^{N}\left[\log\frac{\Gamma\left(\sum_{l=1}^{L}\alpha_{(l)}\right)}{\prod_{l'=1}^{L}\Gamma(\alpha_{(l')})} + \sum_{l=1}^{L}(\alpha_{(l)}-1)\left[\psi\left(\gamma_{2j(l)}\right) - \psi\left(\sum_{l'=1}^{L}\gamma_{2j(l')}\right)\right]\right]
\end{aligned}
\tag{5.6}
$$

[1] $\Gamma(x) = \int_0^\infty e^{-t}t^{x-1}\,dt$

The EM algorithm for inference and parameter estimation is given in Algorithm 7.

Algorithm 7 EM Algorithm for learning in MMBAE

```
Initialize {α_1, α_2, Θ_1, Θ_2, β}
repeat
    E-Step
    Initialize {r_1ij, r_0i, r_2ij, r_0j}
    repeat
        Update {γ_1i, γ_2j} according to Equation 5.5
        Update {r_0i, r_0j, r_1ij, r_2ij} according to Equation 5.5
    until Convergence
    M-Step
    Update {α_1, α_2, Θ_1, Θ_2, β} according to Equation 5.6
until Convergence
```

5.3 Experiments and Results

A preliminary set of experiments were performed with HetRec MovieLens dataset [106] and the results of this model were compared to a global model and a basic PMF model. The dataset was developed in association with GroupLens Research[2], Internet Movie Database(IMDb) website[3] and Rotten Tomatoes review forum [4]. The data was divided into separate training, validation and testing sets, based on the timestamps of the users ratings. The last 4 ratings of each user is used as the test set, the previous 4 ratings were assigned for the validation set and the rest were used in the training set. The processed data consisted of 838694 training ratings, 2113 users, 10053 items, 8452 validation ratings and 8452 test ratings.

For the attributes on this dataset, the user "bias" (the difference between the users mean score and the overall mean of the data) was used as the only user attribute. For

[2]http://www.grouplens.org
[3]http://www.imdb.com
[4]http://www.rottentomatoes.com

Model	Validation RMSE	Test RMSE
Vanilla PMF	0.66847	0.68973
Global Linear Model	0.69036	0.71187
MMBAE (K=2,L=4)	0.68845	0.70943
MMBAE (K=2,L=5)	0.688344	0.70857

Table 5.1: Validation and Test RMSE on MovieLens dataset

item attributes, the average scores and percentages of "fresh" ratings for both critics and audience from Rotten Tomatoes, as well as the movie "bias", were used. The RMSE values for each of the models on the test set is given in Table 5.1.

It is observed that the model beats the global linear model emphasizing the efficacy of clustering. However, the model does not compare with the PMF model. This could be because the predictive model within each co-cluster is a very simple linear model. It is hypothesized that more sophisticated models can lead to better performance. Moreover, the model was developed to generate both affinity values as well as entity attributes, so evaluating the model based on just the affinity ratings gives biased indicators towards the efficacy of the model.

An immediate extension to the work involves evaluating the model towards imputation of the attribute values and assessing the predictive power of the model for cold start scenarios. Further, a more thorough experimentation can lead to insightful conclusions.

Chapter 6

Conclusion and Future Work

This report extensively surveys the state-of-the-art models for recommendation systems that use external side information. The effects of incorporating various forms of side information including entity attributes, temporal attributes and network attributes are analyzed. Further, two models that use side information to improve upon the existing collaborative filtering techniques are proposed.

In *Review Quality Aware Collaborative Filtering*, the novelty of the approach lies in the integration of quality scores based on product reviews, with collaborative filtering to improve the performance of recommender systems. On the one hand, there is a large body of work on the analysis of online product reviews and on the other hand, there is also a fair amount of work in the area of collaborative filtering for recommender systems using various approaches including PMF. This work tries to combine online product review helpfulness with collaborative filtering to improve the overall performance of recommender systems.

In MMBAE, a generative model for affinity estimation is proposed which systematically uses side information available in the form of entity attributes. The simultaneous clustering and prediction paradigm improves the quality of clustering as well the accuracy of the predictive model. Although, the clustering algorithm is limited in predictive accuracy compared to factorization based models, these models provide more interpretable recommendations. Moreover, the model can be easily updated with new ratings as compared to

matrix factorization models. Further, MMBAE is a fully Bayesian generative model which explains both the observed ratings as well as the entity attributes. Apart, from the task of ratings prediction, the model can be used to impute missing values in entity attributes which can then be leveraged for future affinity predictions.

6.1 Future Work

As a part of future work, in *Review Quality Aware Collaborative Filtering*, several additional features including bigrams and semantic features as described in Kim et al. [90] for learning the regression model to predict quality scores, can be incorporated. The other direction of future work involves exploring feature reduction techniques like PCA to reduce the number of bag-of-words features, which can help improve the performance of the regression model. In the current experiments, LDA could not induce latent word distributions that were reflective of quality of reviews. To help improve the performance, supervised approaches like supervised LDA [56] could be explored in future. Further, to overcome the lack of sufficient reviews in data sets, transfer learning approaches [107] could be incorporated for the estimation of quality scores in our framework. Finally, experimental evaluation of the approach on other data sets like the Yelp academic data set and other product categories available in the Amazon data set will also be considered in the future.

Also, there are multiples avenues for extension and improvement in MMBAE. Firstly, the experiments are very preliminary to derive insightful conclusions. More rigorous experimentation on multiple datasets with various number of row and column clusters needs to be performed. Further, the model could be evaluated towards the task of imputing entity attributes and predicting ratings for cold start users. The possibility of incorporating

online update procedure for the MMBAE model needs to be explored. Finally, for both the models, the scalability to large datasets and the possibilities of parallelizations are to be considered.

Bibliography

[1] Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, pages 1–19, 2009.

[2] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, pages 734–749, 2005.

[3] M. Balabanovic and Y. Shoham. Fab: content-based, collaborative recommendation. *Communnucation ACM*, pages 66–72, 1997.

[4] M. Pazzani and D. Billsus. Learning and revising user profiles: The identification of interesting web sites. *International conference on Machine Learning*, pages 313–331, 1997.

[5] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. Grouplens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994.

[6] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Item-based collaborative filtering recommendation algorithms. In *Proceedings of the 10th international conference on World Wide Web*, pages 285–295, New York, NY, USA, 2001.

[7] L. Getoor and M. Sahami. Using Probabilistic Relational Models for Collaborative Filtering. In *Working Notes of the KDD Workshop on Web Usage Analysis and User Profiling*, 1999.

[8] Ruslan Salakhutdinov. Probabilistic matrix factorization. In *Proceedings of Advances in Neural Information Processing Systems 21 (NIPS '08)*, pages 1–8, 2008.

[9] I. M. Soboro and C. K. Nicholas. Combining content and collaboration in text filtering. In *Proceedings of the IJCAI*, pages 86–91, 1999.

[10] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 253–260, 2002.

[11] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Revolutions*, 13(5-6):393–408, 1999.

[12] U. Shardanand and P. Maes. Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 210–217, New York, NY, USA, 1995.

[13] T Hofmann. Learning what people (don't) want. In *Proceedings of the 12th European Conference on Machine Learning (EMCL '01)*, EMCL '01, pages 214–225, 2001.

[14] L. Ungar and D. Foster. Clustering Methods For Collaborative Filtering. In *Proceedings of the Workshop on Recommendation Systems*, 1998.

[15] John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the Fourteenth conference on Uncertainty in artificial intelligence*, pages 43–52, San Francisco, CA, USA, 1998.

[16] Daniel Billsus and Michael J. Pazzani. Learning collaborative information filters. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 46–54, 1998.

[17] Badrul M. Sarwar, Joseph A. Konstan, Al Borchers, Jon Herlocker, Brad Miller, and John Riedl. Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, 1998.

[18] Alon Schclar, Alexander Tsikinovsky, Lior Rokach, Amnon Meisels, and Liat Antwarg. Ensemble methods for improving the performance of neighborhood-based collaborative filtering. In *Proceedings of the third ACM conference on Recommender systems*, pages 261–264, 2009.

[19] Michael Jahrer, Andreas Toscher, and Robert Legenstein. Combining predictions for accurate recommender systems. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 693–702, 2010.

[20] Michael I. Jordan. An introduction to variational methods for graphical models. In *Machine Learning*, pages 183–233, 1999.

[21] Christophe Andrieu. An introduction to MCMC for machine learning. *Machine Learning*, pages 5–43, 2003.

[22] J. Jensen. Sur les fonctions convexes et les inégalités entre les valeurs moyennes. *Acta Mathematica*, pages 175–193, 1906.

[23] Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Information Retrieval*, pages 133–151, 2001.

[24] Nathan Srebro. Weighted low-rank approximations. *Proceedings of 20th International Conference on Machine Learning (ICML '03)*, 2003.

[25] Nathan Srebro and JDM Rennie. Maximum-margin matrix factorization. *Proceedings of Advances in Neural Information Processing Systems 17 (NIPS '05)*, 2005.

[26] JDM Rennie. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of 22nd International Conference on Machine learning (ICML '05)*, 2005.

[27] Yehuda Koren and Robert Bell. Matrix factorization techniques for recommender systems. *Computer*, pages 31–37, 2009.

[28] Steven J. Nowlan and Geoffrey E. Hinton. Simplifying neural networks by soft weight-sharing. *Neural Comput.*, pages 473–493, 1992.

[29] Ruslan Salakhutdinov and Andriy Mnih. Bayesian probabilistic matrix factorization using Markov chain Monte Carlo. In *25th international conference on Machine learning (ICML '08)*, pages 880–887, New York, New York, USA, 2008.

[30] Hanhuai Shan and A Banerjee. Generalized probabilistic matrix factorizations for collaborative filtering. *Data Mining (ICDM), 2010 IEEE 10th*, 2010.

[31] David Chandler. Introduction to modern statistical mechanics. *Introduction to Modern Statistical Mechanics by*, 1987.

[32] Benjamin Marlin. Modeling user rating profiles for collaborative filtering. In *Proceedings of 17th Ann. Conf. Neural Information Processing (NIPS '03)*, 2003.

[33] Benjamin Marlin. The multiple multiplicative factor model for collaborative filtering. In *Proceedings of 21st international conference on Machine learning (ICML '04)*, 2004.

[34] J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association*, pages 123–129, 1972.

[35] Yizong Cheng and George M. Church. Biclustering of expression data. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.

[36] Inderjit S. Dhillon, Subramanyam Mallela, and Dharmendra S. Modha. Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, 2003.

[37] Arindam Banerjee, Inderjit Dhillon, Joydeep Ghosh, Srujana Merugu, and Dharmendra S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *J. Mach. Learn. Res.*, pages 1919–1986, 2007.

[38] L. M. Bregman. The relaxation method of finding the common point of convex sets and its application to the solution of problems in convex programming. *USSR Computational Mathematics and Mathematical Physics*, pages 200–217, 1967.

[39] Thomas George and Srujana Merugu. A scalable collaborative filtering framework based on co-clustering. In *Proceedings of the Fifth IEEE International Conference on Data Mining*, pages 625–628, 2005.

[40] Edoardo M. Airoldi, David M. Blei, Stephen E. Fienberg, and Eric P. Xing. Mixed membership stochastic blockmodels. *Journal for Machine Learning Research*, pages 1981–2014, 2008.

[41] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, pages 993–1022, 2003.

[42] Hanhuai Shan and Arindam Banerjee. Residual bayesian co-clustering for matrix approximation. In *Proceedings of the SIAM International Conference on Data*, pages 223–234, 2010.

[43] Hanhuai Shan and Arindam Banerjee. Bayesian co-clustering. In *Proceedings of the 8th IEEE International Conference on Data Mining (ICDM 2008)*, pages 530–539, 2008.

[44] Pu Wang, Carlotta Domeniconi, and Kathryn Blackmond Laskey. Latent dirichlet bayesian co-clustering. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases: Part II*, pages 522–537, 2009.

[45] Lester Mackey, David Weiss, and Michael I. Jordan. Mixed membership matrix factorization. In *International Conference on Machine Learning (ICML)*, 2010.

[46] A Singh. A unified view of matrix factorization models. In *European conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD '08)*, 2008.

[47] G Takacs, I Pilaszy, B Nemeth, and D Tikk. Scalable collaborative filtering approaches for large recommender systems. *The Journal of Machine Learning Research (JMLR '09)*, pages 623–656, 2009.

[48] Ilya Sutskever, Ruslan Salakhutdinov, and Joshua Tenenbaum. Modelling Relational Data using Bayesian Clustered Tensor Factorization. In *Advances in Neural Information Processing Systems 22*, pages 1821–1828, 2009.

[49] Ian Porteous, Evgeniy Bart, and Max Welling. Multi-hdp: A non-parametric bayesian model for tensor factorization. *Proc. of the 23rd National Conf. on Artificial intelligence, AAAI '08*, 2008.

[50] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommender Systems*, 1999.

[51] Nathaniel Good, J. Ben Schafer, Joseph A. Konstan, Al Borchers, Badrul Sarwar, Jon Herlocker, and John Riedl. Combining collaborative filtering with personal agents for better recommendations. In *Proceedings of the sixteenth national conference on Artificial intelligence and the eleventh Innovative applications of artificial intelligence conference*, pages 439–446, 1999.

[52] Seung-Taek Park, David Pennock, Omid Madani, Nathan Good, and Dennis DeCoste. Naive filterbots for robust cold-start recommendations. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 699–705, 2006.

[53] D Agarwal and Bee-chung Chen. Regression-based latent factor models. *Proceedings of the 15th ACM SIGKDD*, 2009.

[54] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of Royal Statistical Society, Series B*, pages 1–38, 1977.

[55] D Agarwal and Bee-Chung Chen. fLDA: matrix factorization through latent dirichlet allocation. *Proceedings of the third ACM international conference on Web search and data mining, WSDM '10*, 2010.

[56] David M. Blei and Jon D. Mcauliffe. Supervised topic models. In *Advances in Neural Information Processing Systems 21*, 2007.

[57] Aditya Krishna Menon and Charles Elkan. A Log-Linear Model with Latent Features for Dyadic Prediction. *2010 IEEE International Conference on Data Mining*, pages 364–373, 2010.

[58] David M. Blei and John D. Lafferty. Correlated topic models. In *Advances in Neural Information Processing Systems 18*, 2005.

[59] Ian Porteous, Arthur U. Asuncion, and Max Welling. Bayesian Matrix Factorization with Side Information and Dirichlet Process Mixtures. *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI*, 2010.

[60] Deepak Agarwal and Srujana Merugu. Predictive discrete latent factor models for large scale dyadic data. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 26–35, 2007.

[61] Meghana Deodhar and Joydeep Ghosh. Scoal: A framework for simultaneous co-clustering and learning from complex data. *ACM Transactions on Knowledge Discovery and Data Mining*, 2010.

[62] Yehuda Koren. Collaborative filtering with temporal dynamics. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, New York, NY, USA, 2009. ACM.

[63] Liang Xiong, Xi Chen, TK Huang, Jeff Schneider, and J Carbonell. Temporal collaborative filtering with Bayesian probabilistic tensor factorization. *Proceedings of SIAM*, 2010.

[64] PD Hoff. Hierarchical multilinear models for multiway data. *Computational Statistics & Data Analysis*, 2011.

[65] Henry Kautz, Bart Selman, and Mehul Shah. Referral web: combining social networks and collaborative filtering. *Commun. ACM*, 1997.

[66] Manos Papagelis, Dimitris Plexousakis, and Themistoklis Kutsuras. Alleviating the sparsity problem of collaborative filtering using trust inferences. In *Proceedings of the Third international conference on Trust Management*, pages 224–239, 2005.

[67] Chuck Lam. Snack: incorporating social network information in automated collaborative filtering. In *Proceedings of the 5th ACM conference on Electronic commerce*, pages 254–255, 2004.

[68] David Ben-Shimon, Alexander Tsikinovsky, Lior Rokach, Amnon Meisles, Guy Shani, and Lihi Naamani. Recommender system from personal social networks advances in

intelligent web mastering. In *Advances in Intelligent Web Mastering*, pages 47–55. Springer Berlin Heidelberg, 2007.

[69] Francois Fouss, Alain Pirotte, and Marco Saerens. A novel way of computing similarities between nodes of a graph, with application to collaborative recommendation. In *Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence*, pages 550–556, 2005.

[70] Francois Fouss, Alain Pirotte, Jean-Michel Renders, and Marco Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. *IEEE Trans. on Knowl. and Data Eng.*, pages 355–369, 2007.

[71] K. A. R. L. Pearson. The problem of the random walk. *Nature*, 1905.

[72] L. Lovasz. Random walks on graphs: A survey. In *Combinatorics, Paul Erdhos is Eighty*. Janos Bolyai Mathematical Society, 1996.

[73] Paolo Massa and Paolo Avesani. Trust-aware recommender systems. In *Proceedings of the 2007 ACM conference on Recommender systems*, pages 17–24, 2007.

[74] Punam Bedi, Harmeet Kaur, and Sudeep Marwaha. Trust based recommender system for the semantic web. In *Proceedings of the 20th international joint conference on Artifical intelligence*, pages 2677–2682, 2007.

[75] Jennifer Golbeck. Tutorial on using social trust for recommender systems. In *Proceedings of the third ACM conference on Recommender systems*, pages 425–426, 2009.

[76] Zhili Wu, Xueli Yu, and Jingyu Sun. An improved trust metric for trust-aware recommender systems. In *Proceedings of the 2009 First International Workshop on Education Technology and Computer Science - Volume 01*, pages 947–951, 2009.

[77] Yunhong Xu, Jian Ma, Yong-Hong Sun, Jinxing Hao, Yongqiang Sun, and Yongqiang Zhao. Using social network analysis as a strategy for e-commerce recommendation. In *PACIS*. AISeL, 2009.

[78] Hao Ma, Haixuan Yang, Michael R. Lyu, and Irwin King. Sorec: social recommendation using probabilistic matrix factorization. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 931–940, 2008.

[79] Hao Ma, Irwin King, and Michael R. Lyu. Learning to recommend with social trust ensemble. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 203–210, 2009.

[80] Mohsen Jamali and Martin Ester. A matrix factorization technique with trust propagation for recommendation in social networks. In *Proceedings of the fourth ACM conference on Recommender systems*, pages 135–142, 2010.

[81] T. Zhou, H. Shan, A. Banerjee, and G. Sapiro. Kernelized probabilistic matrix factorization: Exploiting graphs and side information. In *Proceedings of SIAM International Conference on Data Mining (SDM)*, 2012.

[82] Risi I. Kondor and John Lafferty. Diffusion kernels on graphs and other discrete structures. In *Proceedings of the ICML*, pages 315–322, 2002.

[83] Lada A Adamic. Zipf, power-law, pareto - a ranking tutorial. Technical report, Information Dynamics Lab, HP Labs, 2000.

[84] B. Marlin, S. Roweis, and R. Zemel. Unsupervised learning with non-ignorable missing data. In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics (AISTATS)*, pages 222–229, 2005.

[85] Benjamin M. Marlin and Richard S. Zemel. Collaborative filtering and the missing at random assumption. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence. 2007*, 2007.

[86] Aditya Krishna Menon, Krishna-Prasad Chitrapura, Sachin Garg, Deepak Agarwal, and Nagaraj Kota. Response prediction using collaborative filtering with hierarchies and side-information. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, 2011.

[87] Deepak Agarwal, Bee-Chung Chen, and Bo Long. Localized factor models for multi-context recommendation. *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11*, 2011.

[88] Ryan Prescott Adams, GE Dahl, and Iain Murray. Incorporating Side Information in Probabilistic Matrix Factorization with Gaussian Processes. *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence*, 2010.

[89] B. Mobasher, R. D. Burke, R. Bhaumik, and C. Williams. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *ACM Transactions on Internet Technologies*, 2007.

[90] S. Kim, P. Pantel, T. Chklovski, and M. Pennacchiotti. Automatically assessing review helpfulness. In *EMNLP–2006*, pages 423–430, 2006.

[91] C. Danescu-Niculescu-Mizil, G. Kossinets, J. Kleinberg, and L. Lee. How opinions are received by online communities: a case study on amazon.com helpfulness votes. In *WWW–2009*, pages 141–150, 2009.

[92] M. P. O'Mahony and B. Smyth. Learning to recommend helpful hotel reviews. In *RecSys–2009*, pages 305–308, 2009.

[93] N. Jindal and B. Liu. Opinion spam and analysis. In *WSDM–2008*, pages 219–230, 2008.

[94] J. Liu, Y. Cao, C. Y. Lin, Y. Huang, and M. Zhou. Low-quality product review detection in opinion summarization. In *EMNLP-CoNLL–2007*, pages 334–342, 2007.

[95] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. In *ACL-HLT–2011*, pages 309–319, 2011.

[96] G. Wu, D. Greene, B. Smyth, and P. Cunningham. Distortion as a validation criterion in the identification of suspicious reviews. In *SOMA–2010*, pages 10–13, 2010.

[97] M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robustness analysis. *ACM Transactions on Internet Technologies*, pages 344–377, 2004.

[98] Z. Wu, J. Cao, B. Mao, and Y. Wang. Semi-sad: applying semi-supervised learning to shilling attack detection. In *RecSys–2011*, pages 289–292, 2011.

[99] D. M. Blei, A. Ng, and M. Jordan. Latent dirichlet allocation. *Journal of Machine Learning Research*, pages 993–1022, 2003.

[100] C.W. Hsu, C.C. Chang, and C.J Lin. A practical guide to svm classification. Technical report, National Taiwan University, 2003.

[101] Thorsten Joachims. Making large scale support vector machine learning practical. In *Advances in kernel methods*, pages 169–184. MIT Press, Cambridge, MA, USA, 1999.

[102] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[103] Y. Low, J. Gonzalez, A. Kyrola, D. Bickson, C. Guestrin, and J. M. Hellerstein. Graphlab: A new framework for parallel machine learning. *CoRR*, abs/1006.4990, 2010.

[104] P. McCullagh and J. A. Nelder. *Generalized linear models*. London: Chapman & Hall, 1989.

[105] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[106] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2nd workshop on information heterogeneity and fusion in recommender systems (hetrec 2011). In *Proceedings of the 5th ACM conference on Recommender systems*, RecSys 2011, 2011.

[107] John Blitzer, Mark Dredze, and Fernando Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *ACL–2007*, pages 187–205, 2007.